

VISIT...

LANZAROTE
Caliente.COM

How to Run Successful Projects III The Silver Bullet

By **David Huxford**





- [Table of Contents](#)

How To Run Successful Projects III Silver Bullet

By Fergus O'Connell

Publisher : Addison Wesley

Pub Date : September 01, 2001

ISBN : 0-201-74806-1

Pages : 352

Slots : 1

Many project managers-especially in software-go their entire professional lives in ignorance of the factors behind the success or failure of their projects.

Permanently in a state of agitation and worry, they just can't explain why some projects work out and others don't.

It doesn't have to be like

this. There is a method underlying all successful projects, and if you follow this method your project is guaranteed to succeed. We have called this method Structured Project Management. The cornerstone of Structured Project Management is the 'Ten Steps'-the first five steps are to do with

planning your project
and the other five with
implementing the plan
and achieving the goal.
*How to Run Successful
Projects III-The Silver
Bullet* builds on the
success of the first and
second editions and
reminds us all, in the
post dot com era, just
how important good
project management

practices are.



- [Table of Contents](#)

How To Run Successful Projects III Silver Bullet

By Fergus O'Connell

Publisher : Addison Wesley

Pub Date : September 01, 2001

ISBN : 0-201-74806-1

Pages : 352

Slots : 1

Copyright

PREFACE TO THE NEW EDITION

PREFACE TO THE SECOND EDITION

Acknowledgments

And finally, a note on sexist language

PREFACE TO THE FIRST EDITION

ABOUT THE AUTHOR

INTRODUCTION

The Silver Bullet

Permission acknowledgments

Part 1: ANALYZING AND PLANNING F

Chapter 1. STEP 1: VISUALIZE THE PRIZE

INTRODUCTION

IDENTIFYING THE GOAL

DEFINING THE GOAL

THE REASON FOR THE GOAL

MOTIVATING THE TEAM

CHANGES TO THE GOAL AND CH

APPLICATION TO SOFTWARE ENGINEERING PSI CONTRIBUTION

Chapter 2. STEP 2: MAKE A LIST OF JOBS INTRODUCTION

MAKING A CHECKLIST

IDENTIFYING THE JOBS WITH FUNCTIONAL CODE

APPLICATION TO SOFTWARE ENGINEERING PSI CONTRIBUTION

Chapter 3. STEP 3: THERE MUST BE A MODEL INTRODUCTION

A ROLE MODEL

APPLICATION TO SOFTWARE ENGINEERING PSI CONTRIBUTION

Chapter 4. STEP 4: ASSIGN PEOPLE TO JOBS INTRODUCTION

EACH JOB HAS A NAME

PEOPLE'S OTHER COMMITMENTS

MONOLITHIC TEAM OR FLAT STRUCTURE

HIERARCHY OR TEAM STRUCTURE

MAXIMIZE STRENGTHS

ASSIGNING PEOPLE TO JOBS

ASSIGNING PEOPLE TO JOBS WITH

APPLICATION TO SOFTWARE ENGINEERING

PSI CONTRIBUTION

Chapter 5. STEP 5: MANAGE EXPECTATIONS
ERROR, HAVE A FALLBACK POSITION

INTRODUCTION

CONTINGENCY: ALLOW A MARGINAL
POSITION

MANAGE EXPECTATIONS

A WORD ON COMMITTING

APPLICATION TO SOFTWARE ENGINEERING

PSI CONTRIBUTION

Part 2: REVIEWING AND IMPLEMENTING
GOAL

Chapter 6. STEP 6: USE AN APPROACH

INTRODUCTION

THE LAZY PROJECT MANAGER

APPLICATION TO SOFTWARE ENG

PSI CONTRIBUTION

Chapter 7. STEP 7: KNOW WHAT'S

INTRODUCTION

USING YOUR PLAN AS INSTRUMI

MANAGER'S DAY

POSITIVE SIGNS

APPLICATION TO SOFTWARE ENG

PSI CONTRIBUTION

Chapter 8. STEP 8: TELL PEOPLE V

INTRODUCTION

STATUS REPORTS

THE LAZY PROJECT MANAGER'S

A VARIATION ON THE LAZY PROJ

APPLICATION TO SOFTWARE ENG

PSI CONTRIBUTION

Chapter 9. STEP 9: REPEAT STEPS

INTRODUCTION

WHEN SHOULD WE UPDATE THE
APPLICATION TO SOFTWARE ENG
PSI CONTRIBUTION

Chapter 10. STEP 10: THE PRIZE
THE PRIZE
THE RECKONING
PSI THRESHOLDS
APPLICATION TO SOFTWARE ENG
PSI CONTRIBUTION

Part 3: RUNNING MULTIPLE PROJECT

Chapter 11. THE LAZY PROJECT M
INTRODUCTION
PROJECT MANAGER'S MONTHLY

Chapter 12. PROJECT MANAGER'S
INTRODUCTION
SPECIFIC WEEKLY JOBS

Chapter 13. PROJECT MANAGER'S

INTRODUCTION

Part 4: HOW TO ASSESS PROJECT P

Chapter 14. ASSESSING PROJECT

INTRODUCTION

FIRST LEVEL CHECKS

SECOND LEVEL CHECKS

THIRD LEVEL CHECKS

GUIDELINES FOR WRITING PROJ

Part 5: THE REST OF THE WHEREWI

Chapter 15. RESOLVING ISSUES: | MAKING

INTRODUCTION

PROBLEM-SOLVING METHOD

Chapter 16. COPING WITH STRES

INTRODUCTION

WAYS TO REDUCE STRESS

Chapter 17. PICKING THE RIGHT F

INTRODUCTION

METHOD OF INTERVIEWING

INTERVIEW QUESTIONS

Chapter 18. NEGOTIATION

INTRODUCTION

PRINCIPLED NEGOTIATION

Chapter 19. MEETINGS

INTRODUCTION

THE MEETING ALARM

ORGANIZING AND RUNNING MEI

Chapter 20. PRESENTATIONS

INTRODUCTION

Chapter 21. SHORTENING PROJEC AND DESIGN

INTRODUCTION

WHAT TAKES THE TIME?

HOW TO CARRY OUT AN AAD

RISKS IN HOLDING AN AAD

PRACTICAL CONSIDERATIONS

AFTERWORD

DELEGATION (OR THE REAL JOY OF)

APPENDICES

Appendix 1. ISO 9000 ESTIMATING

INTRODUCTION

Section 1. WORK BREAKDOWN

DEPENDENCIES

Section 2. AVAILABILITY OF RES

Section 3. THE PROJECT MODEL

Section 4. BUILD IN CONTINGEN

Section 5. IDENTIFY OPTIONS

Section 6. THE PREFERRED OPT

Section 7. SAMPLE WBS

Appendix 2. STRUCTURED PROJECT **AND METHODOLOGIES**

INTRODUCTION

Appendix 3. PROBABILITY **OF SUCCESS**

INTRODUCTION

CALCULATING A PROJECT'S PSI

HOW TO CALCULATE THE PSI

Appendix 4. BASIC PRECEPTS AND

INTRODUCTION

THE FOUR BIG ONES

ABBREVIATIONS AND RULES OF

CRITICAL PATH

GLOSSARY OF GENERAL PROJECT

Appendix 5. ADDITIONAL FORMS

THE ESTIMATING SCORE CARD

THE CHANGE REQUEST FORM

CHANGE REQUEST LOG

Appendix 6. LEARNING **MICROSO**

INTRODUCTION

MODULE 1: BASICS OF PROJECT

MODULE 2: GETTING STARTED V

MODULE 3: MENUS OF MS PROJ

MODULE 4: SETTING DEFAULTS

MODULE 5: CREATING TASKS

MODULE 6: ENTERING TASK DUF

MODULE 7: USING HELP

MODULE 8: USING VIEWS

MODULE 9: SETTING TASK DEPE

MODULE 10: USING ORGANIZAT

MODULE 11: OUTLINING TASKS

MODULE 12: PRINTING VIEWS

MODULE 13: ASSIGNING RESOU

[MODULE 14: OPTIMIZING THE S](#)

[MODULE 15: RESOURCE LEVELI](#)

[MODULE 16: USING THE BASELI](#)

[MODULE 17: UPDATING TO REFL](#)

[MODULE 18: MULTIPLE PROJECT](#)

[MODULE 19: USING REPORTS](#)

[REFERENCES AND FURTHER READI](#)

[References](#)

[Further reading](#)

Copyright

PEARSON EDUCATION LIMITED

Head Office:

Edinburgh Gate

Harlow CM20 2JE

England

Tel: +44 (0)1279 623623

Fax: +44 (0)1279 431059

London Office:

128 Long Acre

London WC2E 9AN

Tel: +44 (0)20 7477 2000

Fax: +44 (0)20 7240 5771

Websites: www.informit.uk.com

www.aw.com/cseng

First published in Great Britain
in 2001

© Pearson Education Limited
2001

The right of Fergus O'Connell to

be identified as the author of this Work has been asserted by him in accordance with the Copyright, Designs and Patents Act 1988.

British Library Cataloguing in Publication Data

A CIP catalogue record for this book can be obtained from the British Library.

Library of Congress Cataloging in Publication

Data

Applied for.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without either the prior written permission of the Publisher or a licence permitting restricted copying in the United Kingdom issued by the Copyright Licensing Agency

Ltd, 90 Tottenham Court Road, London W1P 0LP. This book may not be lent, resold, hired out or otherwise disposed of by way of trade in any form of binding or cover other than that in which it is published, without the prior written consent of the publisher.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Pearson Education Limited has made every attempt to supply trademark information about manufacturers and their

products mentioned in this book. Microsoft Project 2000® and Microsoft Office® are trademarks of Microsoft Corporation.

10 9 8 7 6 5 4 3 2 1

Designed by Claire Brodmann
Book Designs, Lichfield, Staffs.

Typeset by Pantek Arts Ltd,
Maidstone, Kent.

Printed and bound in Great
Britain by Biddles Ltd of
Guildford and King's Lynn.

The Publisher's policy is to use paper manufactured from sustainable forests.

Dedication

THIS BOOK IS DEDICATED TO
MY FRIEND, **JIM FALLON**,
WHOSE TRAGIC DEATH
ROBBED THE WORLD OF A
TRULY GOOD PERSON

PREFACE TO THE NEW EDITION

I'm delighted that Addison-Wesley has chosen to re-issue this book. The origins and evolution of the book are amply described in the prefaces to the first and second editions, and there is nothing I can usefully add to these.

The book originally saw the light of day thanks to Viki Williams, the best editor in the world. Incredibly, eight years

later, Viki is back on the case and this edition is once again the result of her hard work and belief. Susan Harrison did such an extraordinarily meticulous job on the preparation of the book for publication that she should really be credited as a co-author. Thanks, Susan.

Ireland, May 2001

PREFACE TO THE SECOND EDITION

If you are in the business of writing books, then one of the things you spend a lot of your time doing is in looking for the opening line. There are many reasons for this, but perhaps the most selfish one is that, unless you are very prolific, you don't get to do opening lines very often. In my own case it is three years since I did my last one, and goodness knows how long it will be

before I get to do another.

[Tom DeMarco and Timothy Lister \(1987\)](#) found their first line in the classic, *People Ware*: "Somewhere today," their memorable opening runs, "a project is failing." And today, nine years later, despite everyone's best efforts, projects are still failing. The French General Ducrot put things more colorfully after the defeat of the French forces at the Battle of Sedan, when he said "*Nous sommes dans un pot de chambre, et nous y serons emmerdés.*" And today, even as

I write or as you read,
somebody is waking up to find
that they are indeed in a *pot de
chambre*, and that they are
very much *emmerdés*.

It doesn't have to be like this.
That was the message of the
first edition of this book, and it
is more than ever the message
now.

I had thought that I was
finished with *How To Run
Successful Projects*, that there
was not much more that
needed to be said, but the
intervening years have proved

me wrong.

By far the most significant thing that I have learned is that what I knew in 1992 to be true, others can now independently confirm: that is, that the Ten Step Method is *the* solution to the software industry's biggest problem that of software projects not coming in on time, within budget or delivering what was required. I knew this three years ago when I wrote the first edition, but I didn't think the world was quite ready for a claim like that then. Now, I can

come out of the closet and make the claim unashamedly, and secure in the knowledge that it can be backed up by people who have actually used it. Of course, in a sense, this should come as no surprise because of the origin of the Ten Steps. Since the Ten Steps were derived from understanding why some projects succeed and others fail, the fact that they work for software projects merely confirms what I have thought for years, that despite all the propaganda to the contrary

software projects are no different from any other kinds of projects.

I have learned something else. The Ten Steps don't just guarantee a successful project, they are also the minimum that has to be done *for* a successful project. Mathematicians would call the Ten Steps a necessary and sufficient condition for a successful project — you have to do what the Ten Steps require, but having done this much, that is all you have to do. To put it differently, the Ten Steps tell you when you have

done enough project management. Your days of waking in the middle of the night and wondering if you are doing enough are over once you start applying the Ten Steps.

This property of the Ten Steps opens up the wonderful vista that we have called the Lazy Project Manager. *Lazy* is normally used as a derogatory term, but here we mean it to be highly complimentary. The Lazy Project Manager does the minimum possible and always has a successful outcome to

the bewilderment of her colleagues.

In detail, the main changes you will find in this edition of the book are the following:

- Since the Ten Steps guarantee success, we have tried to make it as easy as possible for you to begin applying them. Many of the things we provide here are things that our clients have asked us for over the last three years.

- The Probability of Success Indicator (PSI), rather than being a bit of a footnote, is now integral to the Ten Steps. Each step's contribution to the PSI is explained more fully. We can do this because we have managed to apply the PSI to several thousand projects since the first edition of the book appeared.
- Software estimation has been simplified and made much more usable. Looking

back on it I realize that I was confused about software estimation. I still saw it as something which was different somehow from estimating in other disciplines. Now, I realize it is not at all.

- Time management has been integrated with the Ten Steps. This is because we now want you to be not just a successful project manager but a *Lazy* Successful Project Manager.

Acknowledgments

In the evolution of this material over the last three years, I have learned much from many people, and there are some I would particularly like to acknowledge.

Dee Carri and Mary Martin at Elan Corporation; Michael Rice and Colm Quinn of the Kerry Group; Paul O'Dea and Paul Fox of Credo Group; Dermot Gilson of Avid Technology; Tony Mulqueen at ISOCOR; Allan

Young from Scottish Equitable in Edinburgh; Sacha Jovanovic of MPA Stuttgart; Guido Haesen and Peter Lowe of the European Commission in Luxembourg; my former bosses at Retix, Tony Fonzé and Ron Rudolf; Russell Altendorff of Travelex Corp.; Jan Middleton of SSA Europe; Tim Greening-Jackson of KNX; Paul Renahan of ITP; Dermot Bolger and Redmond Sweeny at Telecom Ireland Software; and John O'Leary of BP Exploration in Colombia. All, whether they know it or not, helped me to see connections

between things and gain new insights.

The material in [Appendix 1](#) first appeared in an article entitled "How to become a world class estimator" in Issue 4 of the *Journal of Structured Project Management*. It was then further extended as part of an internal ETP project. However, I would like to acknowledge a number of contributions to its final shape from Telecom Ireland Software, a company which uses this procedure.

Sean McEvoy and David

Conway prepared the Guide to MS Project included in [Appendix 6](#).

The book is written, and then a small group of skilled, dedicated and talented people bring it to you. There are the production people, in my case, Ann Greenwood, Louise Wilson (for the last time!) and Val Jones. There are the editors three did time on this book: Viki Williams who did her usual highly professional job, was a pleasure to work with and then, just as I handed in the manuscript, upped and left.

Good luck on your travels Viki. I guess I'll never get that £2 now! Christopher Glennie who held the fort; Jason Dunne, who was the best of project managers enthusiastic, helpful, creative, supportive and (best of all) gave me deadlines I could meet! (If only he'd give me that screen saver. I'd really be happy!!)

There are the cover designers Maurice Farrell, John O'Regan and Martin Swords. Who knows how many magazines they exhausted trying to get those holes just

right!

Myles Dunne, technical author to the gentry, prepared the camera-ready copy with his customary skill and made a book out of a heap of paper, diskettes and garbled notes from me.

Also, the show's not over until the book lands in your lap. For this I have to thank the sales people: Alan, Craig, Delya, Karen, Lesley, Wendy and Sam. I know that books don't just sit there and sell themselves it's the reps who have to get out

and push them. I sell myself, so I know how tough it can be. Thank you for all your hard work on my behalf. It was your commitment and dedication that made the first edition a success.

And finally, there are Hugh and Bernadette. Hugh wasn't old enough to read the first edition, so he had an excuse, Bernadette had none! I'm sure they'll both get endless hours of reading pleasure out of this one!!

And finally, a note on sexist language

This is one area where you can't win. To use *he* and *his* throughout, even with a disclaimer that no offense is intended, doesn't wash with some people. To do the opposite and use *she* and *her* throughout, one is accused of tokenism or of treating seriously something that should be regarded as trivial.

Whatever you choose to do, you can be guaranteed you'll

offend or irritate somebody.
Sigh!

Here's the policy I have adopted. I have used *he/his* and *she/her* as and when the mood has taken me! I have done this because the juxtaposition of the feminine and masculine sometimes (a) surprises people and (b) conjures up different mental pictures.

Ireland, March 1996

PREFACE TO THE FIRST EDITION

In 1979 I bought [Tom De Marco's book \(1978\)](#) *Structured Analysis and System Specification*. At the time, I had been working for about three years as a programmer, but had ambitions to be a systems analyst. Those in the know, while applauding my ambition, made it clear that not everyone was cut out to be a systems analyst. It was highly likely that I might not have the "right

stuff" that particular combination of personality, intellect and way of looking at things that characterized the systems analyst. In short, the thinking went, systems analysis was very much an art, and in the same way that not everybody was born to be a great pianist, not everybody could be a systems analyst.

After reading De Marco's book, it would not be an exaggeration to say that the scales had fallen from my eyes. Or to put it more bluntly, I was gobsmailed! The great

mystery of systems analysis was laid bare. It wasn't an art not much anyway it was a method, an approach, a technique. While it would be a wild oversimplification to say that anyone could do it, it certainly wasn't sorcery or genius. The book presented a method and if you carried out the method you were doing systems analysis. Systems analysis has long since ceased to have any mystique about it, so now we software practitioners have had to find another black art, another

mystery with which to clothe ourselves. That black art is project management, and you only have to look at the success statistics for projects to see why the world at large might perceive it to be so. The premise of this book is that project management is no more a black art than systems analysis is. Project management too can be described by a method, and that is what this book is primarily about. De Marco's book had a seven-step method; this one has a ten-step method.

The method is derived from studying other projects, why some succeed and others fail.

The argument is simple: if you do enough of the things on your project that have made other projects successful, the chances are yours will be too; and, conversely, if you engage in practices which have been used on failed projects, then God help you.

[Parts 1](#) and [2](#) of the book contain the Ten Steps Method, and are intended to be read sequentially. Other than that,

you can dip in as you please. While the book uses the software industry to illustrate many of its ideas, the Ten Steps Method isn't unique to software. It can be applied to any venture, in any discipline; and is equally applicable to business or personal projects. Indeed, one person who attended one of my company's courses said that it changed his life. While it may not do that for you, I can pretty much guarantee you will learn new things from it.

Project management is

concerned with bringing about change, about making things happen. In a world where we are in need of much change and this is true of my own small country as much as anywhere it is the project managers who will make the visions come true. Bad project management results in a colossal waste of the world's resources. There is enough of that going on, and we should be trying not to add to it.

But more than anything else, project management is *fun*. Software people tend to think

that all the fun has gone out of their lives when they are promoted from "doing technical work" to "managing people." Nothing could be further from the truth. For infinite variety, constant challenge, unpredictability, adrenalin-burn and ultimate sense of achievement, project management is hard to beat; and if this book succeeds in transferring some of that enthusiasm to you it will have been worth the struggle it took to write it.

Many people customers,

students, colleagues, peers, bosses, both current and former contributed, consciously and sometimes unwittingly, to this book. Rather than changing the names to protect the innocent I have chosen to omit all names. However, two groups of people deserve particular mention. One is those who read and commented on drafts of the book. The other is those who attended courses on the content while the material was still in its formative years and growing from the "EOTP

Methodology" through "ETP" to its current simple name of Structured Project Management. There are some people and organizations that I would like to name: Telecom Ireland Software (TIS) gave me permission to use some of the examples in the text, for which I am very grateful. In addition, Tom Moylan and Eugene Smyth, both of TIS, gave me the opportunity to apply and thereby refine many of the ideas in the book.

Beaumont Hospital were kind enough to allow me to use an

example, which one of their staff, Cathy Keany, put together.

Whether they know it or not, Ray Welland of the University of Glasgow and Pat O'Reilly of Siemens Nixdorf both encouraged me at points where I had begun to despair about the material ever seeing the light of day. Often it is small incidents that change lives, and conversations which these people may have forgotten, were crucial in the evolution of the material as well as steadying my will to continue.

As she knows, Viki Williams of Prentice Hall will have a permanent place in my heart for having given me my first break. Viki still owes me £2 from the time we first met, but it's OK Viki buy me a pint next time you're in Ireland!

Myles Dunne put the finished product together, ironing the crumpled pages, drying the soggy ones, etc. and takes credit for its final appearance.

Bernadette McHugh put up with (relative) poverty and all that it entails, while this was being

written. Thanks Bernie for
staying with it and me!
Fortitudinem vincimus.

Hugh O'Connell and some
(furry) friends of his were right
there with me when I wrote
much of the book, as is
evidenced from old drafts that
have muddy paw-prints or
coloured scribblings on them.
Thanks guys you may not
have made it easier, but it was
more fun!

Ireland, January 1993

ABOUT THE AUTHOR

Fergus O'Connell graduated with a First in Mathematical Physics from University College Cork, Ireland. His 25 years' experience — 22 of those in project management positions — cover commercial data processing, microprocessor-based office automation systems, computer networking, data communications and telecommunications. He has worked with companies such as

CPT, ICL and Retix, before founding ETP in 1992. ETP is Ireland's largest project management company specializing in knowledge and high-tech industries. Fergus is currently Chairman of ETP. His experience covers projects in Australia, Britain, Denmark, Germany, Ireland, Luxembourg, Sweden, Switzerland and the United States. He has taught project management on three continents.

Fergus is the author of three books on project management: *How To Run Successful Projects*

II The Silver Bullet (Prentice Hall, 1996), *How To Run Successful High-Tech Project-Based Organizations* (Artech House, 1999) and *How To Run Successful Projects In Web-Time* (Artech House, 2000). The first of these, sometimes known simply as "The Silver Bullet," has become both a bestseller and a classic. This year, 2001, will see the publication of two further books: *Managing e-Projects for Rapid Business Value* (Addison-Wesley) (co-authored with Professor John Brackett) and

Simply Brilliant: The Competitive Advantage of Common Sense (Prentice Hall).

Fergus has written on project management for *The Sunday Business Post*, *Computer Weekly* and *The Wall Street Journal*. He has lectured on project management at University College Cork, Bentley College, Boston University and on television for the National Technological University.

He lives with his wife, son, daughter, three dogs, two

horses, three ponies and a donkey, beside the River Barrow in Ireland.

INTRODUCTION

[The Silver Bullet](#)

[Permission](#)
[acknowledgments](#)

The Silver Bullet

In 1987, along with most of the rest of the software industry, I read Fred Brooks' famous article *No Silver Bullet* (1987). (For those of you who don't know it must be my age because there was a time when everyone knew who he was Fred Brooks is the doyen of software project managers, the author of that most famous of project management tracts, *The Mythical Man-Month*.)

Brooks' point was simple: despite all the advances in hardware and software technology, there seemed to have been no corresponding *"development in either technology or management technique"* [my italics] that would enable IT people to bring projects in on time, within budget and deliver what was required. Brooks said it and the world agreed, not because it was Brooks, though I'm sure that helped, but more because that seemed to correspond to our experience as well. Brooks

saying it merely put the seal of approval on it.

So where am I coming from? In calling this book *The Silver Bullet*, have I discovered the elusive breakthrough? Failing that, have I taken leave of my senses? Or more serious still, am I starting to believe my own marketing material?? Most important of all, do I actually have The Silver Bullet?

Well, the answer is yes and no. No, in the sense that I don't have a development in technology or management

technique, some nifty new algorithm, or management technique extracted from the writings of an obscure 12th century philosopher. Sorry to disappoint you on this score.

But you see, I think Brooks and all the rest of us have been barking up the wrong tree. We were looking for a *development*. Because computers and software were new, the thinking went, we needed some new thing to manage them. For some reason, it never occurred to us that *old* things might work.

Perhaps you might allow me to digress here for a moment, because I believe I know how this thinking originated.

People of my generation people who remember mainframes and punched cards and the term data processing will remember what it was like in those early days (I am thinking of the late sixties/early seventies). I remember the Computer Center when I was still at college: the air-conditioned room, the machine with flashing lights, and the busy

acolytes who tended it sharp-suited IBM systems engineers or the more casually dressed, but no less boffin-like, Computer Center personnel. Sometimes the machine went down or a new operating system release had to be installed, and when we students asked how long it would take, we were told "We don't know. It'll take as long as it takes."

We computer science students longed to join the ranks of the boffins. To learn the new technology. To tend the

machine. To be able to say to those who didn't understand the new technology: "We don't know. It'll take as long as it takes."

And so the myth was born. *La Grande Illusion*, I call it. La Grande Illusion maintains that IT (and especially software) projects are difficult, if not impossible, to estimate. Largely as a result of this, IT projects enjoy the notorious track record that they do.

La Grande Illusion is a fundamental article of faith of

the software industry. It's reasoning goes like this: "Software entities are more complex for their size than perhaps any other human construct." (The quote is from our old friend Brooks again ([Brooks, 1975](#))). Therefore, all bets are off where estimating software is concerned. Things that would make sense in other areas don't make sense in software.

Thus software people will do things like:

- Agree a fixed-price contract for something that isn't yet specified; a contract that may well have penalty clauses for late delivery.
- Do plans and schedules based on people working 7-day weeks, 15 hours a day.
- Do realistic plans but then back down on them at the first sign of resistance from their management or a customer.
- Not try to estimate

anything at all on the basis that software estimating is only for wimps.

And they will do all of these things without batting an eyelid all because of the basic article of faith of the software industry which says that software projects are difficult if not impossible to estimate.

(Where else would you find such a career: technically challenging, relatively highly paid, and where, ultimately, we can shrug and say, "Hey, how

could we have known, things take as long as they take"?)

A whole industry has risen up about this article of faith. In Europe, even as we speak, there are research programs exploring the outer reaches of the problem. I have numerous books — one is nearly 800 pages long, an immense work of scholarship — that propose ways of dealing with it. Around the world, millions, if not billions of research dollars are being spent in the search for the development Brooks spoke about — a way to accurately

estimate software projects.

And all of this, I believe, is barking up the wrong tree. Because the answer was there waiting for us all the time.

Attempts successful to some extent have been made to compare the building of software to the construction industry. I think, however, a much better comparison is to compare the creation of software to the making of a film. Both take a written vision a script in the case of a film, a specification in the case of

software and convert it into images on a medium (frames on celluloid or bits on disk or tape).

Now the film industry contains noteworthy examples *Revolution*, *Heaven's Gate*, *Waterworld*, to name but a few of films that have gone ballistically over budget and schedule. But these days, films are shot in a businesslike and highly predictable way to meet specified budget, schedule and quality targets.

And what is the key to this?

The key is the pre-production phase. In the book, *My Indecision is Final*, Jake Eberts ([Eberts and Ilott, 1990](#)) describes Sir Richard Attenborough's pre-production phase for the film, *Gandhi*:

He [Attenborough] had to shoot the film during the cool season in India, starting in November and finishing the following April or May. (The summer would simply be too hot: a film crew could not function in 110 degree

heat.) But to start in November he had to be making preparations now, six months ahead: hiring the cast and crew, building the sets, sorting out the costumes, getting all the permissions needed to shoot in India, shipping the equipment and so on. These tasks, known as pre-production, are fairly straightforward if you are shooting in your own country, but are horrendously complicated when you are shooting

overseas. To take a simple example, if you are going to fly in 125 people to make a movie, you have to book the hotel rooms and pay for them, or at least put down some sort of deposit, well in advance. *That means knowing now exactly where you will want each of those 125 people to be on any given day over the four or six months that the film will be shooting.* [my italics]

If you replaced the 125 people with the size of your particular software team, and replaced the phrase "film will be shooting" with "software will be under development," then, essentially, you have the Silver Bullet. And this Silver Bullet didn't come from any *development* it came from well-tried techniques that have evolved over the hundred or so years of the film industry, which in turn have come from projects in other disciplines and areas of technology.

This then is the Silver Bullet.

The premise of this book is simple.

If you do the things I've described in this book on your projects, then your projects will always be successful. No ifs, ands, buts, maybes, provideds, or anything else.

This book is about a method called Structured Project Management. Structured project management is the result of over 25 years of research into why some projects succeed and others

fall. While structured project management isn't tied to any particular kind of project, this book focuses a lot on IT and software projects, areas notorious for failures.

Software people will find that structured project management bears the same relationship to conventional project management that:

- structured analysis & design bears to conventional analysis and design;

- structured programming bears to programming;

that is, it replaces a process which is highly individualistic, with a process that is standard, repeatable, predictable and consistent. We are often asked if structured project management is a "methodology" software people love the notion of a methodology and the answer is yes and no. In their wonderful book, *People Ware*, [Tom De Marco and Timothy Lister \(1987\)](#) define two

versions of the word methodology: "methodology" and "Methodology." Methodology with a big "M" is defined as "a general systems theory of how a whole class of thought-intensive work ought to be conducted. It comes in the form of a fat ('a linear foot or more of shelf space') book that specifies in detail exactly what steps to take at any time, regardless of who's doing the work, regardless of where or when."

If your question is whether structured project management

is such a methodology, the answer is definitely not.

Methodology with a small "m" is a different fish. Methodology with a small "m" is defined as a "basic approach one takes to getting a job done. It doesn't reside in a fat book, but rather in the heads of the people carrying out the work."

Structured project management is a methodology with a small "m."

Though we will deal primarily with software projects in this book, structured project

management states that any human endeavor or venture can be treated as a project, and the book could then be used by anybody responsible for leading such endeavors or ventures. Three types of readers are envisaged:

1. Project managers managing their first project;

- Seasoned project managers who would benefit from the use of a formal project management method;

- Seasoned project managers who feel the need for a top-up in their skills; a refresher course; a fine-tuning on how they do things; or some kind of reference point against which to judge their projects.

The book is divided into five parts. [Parts 1](#) and [2](#) present the Ten Steps of structured project management. [Parts 1](#) and [2](#) show you how to run any project successfully.

Most of us don't have the luxury of being able to

concentrate all of our efforts on a single project. Most of us are involved in many projects simultaneously, and so [Part 3](#) shows how to run multiple concurrent projects.

You will see as the book unfolds that most project disasters can be predicted long before they happen. The Ten Steps prove that you pretty much make or break a project during the planning phase. To enable you to assess project plans quickly and effectively, we have come up with an approach, based on the Ten Steps, which enables

you to do this. This is described in [Part 4](#).

[Part 5](#), called The rest of the wherewithal, presents a series of chapters on subjects of which the project manager needs to be aware. Many of the subjects are those covered in other management books or on training programs. There are chapters on:

- Problem solving and decision making
- Stress management

- Picking the right people
- Negotiation
- Meetings
- Presentations
- Shortening projects through accelerated analysis & design.

Finally, there are a number of appendices which are referred to in the text.

With many organizations now

moving away from traditional management structures (hierarchical, matrix, etc.) to project-based approaches, it is hoped that this book will become seminal in advancing research in this critical area. The emphasis throughout the book is on:

- simplifying and getting to the heart of any issue
- useful and practical ideas
- presenting ideas in an entertaining and

stimulating way.

Permission acknowledgments

The author is grateful to the following for permission to reproduce excerpts from works indicated.

A.P. Watt Ltd, on behalf of Roland Huntford for permission to quote from *Scott and Amundsen* by Roland Huntford.

Verso for permission to quote from *Judgement*

Over the Dead by Trevor Griffiths.

Unwin Hyman, of
HarperCollins Publishers
Ltd, for permission to
quote from *The Lord of
the Rings* by J.R.R.
Tolkien.

Excerpt from *The
Fellowship of the Ring* by
J.R.R. Tolkien. Copyright
© 1954, 1965 by J.R.R.
Tolkien. Copyright ©
renewed 1982 by
Christopher R. Tolkien,

Michael H.R. Tolkien, John F.R. Tolkien and Priscilla M.A.R. Tolkien. Reprinted by permission of Houghton Mifflin Co. All rights reserved.

Bantam Doubleday Dell Publishing Group Inc. for permission to quote from *Voices of Freedom* by Henry Hampton and Steve Frayer.

After some time they crossed the Water, west of Hobbiton, by a narrow

plank-bridge. The stream there was no more than a winding black ribbon, bordered with leaning alder trees. A mile or two further south they hastily crossed the great road from the Brandywine Bridge; they were now in the Tookland and bending south-eastwards they made for the Green Hill Country. As they began to climb its first slopes they looked back and saw the lamps in Hobbiton far off twinkling in the gentle

valley of the Water. Soon it disappeared in the folds of the darkened land, and was followed by Bywater beside its grey pool. When the light of the last farm was far behind, peeping among the trees, Frodo turned and waved a hand in farewell. 'I wonder if I shall ever look down in that valley again,' he said quietly.

The Lord of the Rings

J. R. R. Tolkien

Late in the evening on June 6th, Amundsen came out of his house, shut the door behind him, leaving things as if returning in an hour or two, walked briskly through the trees, and boarded Fram. There was a rattle of chains, the anchor was raised, and slowly she swung out into the fjord. 'Sailed at midnight', ran the opening entry of Amundsen's diary ...

Scott and Amundsen

Roland Huntford

One writes in a certain sort of way because one is a certain sort of person; one is a certain sort of person because one has led a certain sort of life.

A. A. Milne

Part 1: ANALYZING AND PLANNING PROJECTS

THE NATURE OF PROJECTS

This book deals primarily with software development projects, although much of what is said here will be applicable to projects in any discipline. In particular, the definition of

"project" that I am going to use is that any combination of a noun and verb together constitute a project. Thus, using this definition, any of the following could be a project:

- be the first person at the South Pole
- get a promotion and a raise
- achieve record sales of a new product

- set up a new business division
- cost-reduce a manufacturing process
- build an oil refinery
- put a man on the moon
- develop a new computer system
- build the A-12 airplane

Note that this definition is recursive: it defines the thing in terms of itself. For example, the project "develop a new computer system" in itself consists of a number of projects:

- develop the hardware
- develop the software
- integrate the hardware and the software
- release the system

Each of these projects can be further broken down. For "develop the software" we might have:

- identify the requirements
- do the high-level design
- do the functional specification
- etc.

The key factor about

projects is that they have a birth, a life and a death. They are not ongoing sorts of things. Identifying the birth, life and death particularly the death are key to the success of a project. In the course of this book I will use various analogies when talking about projects.

The project as a journey

A project is like a journey;

it involves identifying a destination, setting out, traveling and ending up somewhere hopefully the place you intended to be.

The project as a state change

A project is all about achieving some goal. The universe is in one state before the project, a different state afterwards, and the difference is the

goal. In the examples given previously the destinations/goals/state changes are:

- a person has stood at the South Pole
- you have received the promotion and raise
- you have achieved record sales of the new product
- the new business division is up and

running

- the cost of the manufacturing process has been lowered
- the oil refinery has been built
- a man has landed on the moon
- the new computer system is available
- the A-12 is operational

This book presents a methodology called Structured Project Management for handling projects. In the terminology used above, structured project management will help you to identify where you want to go. It will then give you the wherewithal to plan your journey, carry out the journey, arrive at your destination. To put it another way, you will achieve your goal.

STRUCTURED PROJECT MANAGEMENT: THE TEN STEPS

The Ten Steps are the cornerstone of structured project management. They are covered in the next ten chapters of this book.

The Ten Steps are:

- 1. Visualize the goal;
set your eyes on the
prize;**

- Make a list of jobs to be done;
- There must be one leader;
- Assign people to jobs;
- Manage expectations, allow a margin for error, have a fallback position;
- Use an appropriate leadership style;
- Know what's going on;

- Tell people what's going on;
- Repeat Steps 1 – 8 until step 10; and finally
- The Prize.

I said in the first edition that the first five steps were to do with planning your project, the other five with implementing the plan and achieving the goal. While this is still one hundred percent true, I realize now that there is a

deeper, more intuitive reason as to why the steps are structured in this way.

Steps 1–5 do indeed produce a plan for your project. However, what they also do is to define one possible way in which the project might actually unfold. Let me put this another way. If you build into your project plan the level of intricate detail I am going to ask of you in Steps 2 and 4, then your plan will reflect a possible way that the project could

actually happen.

The key to all of this is the word "detail" above. The conventional wisdom is that you can't know much about a project, particularly a software project, at the beginning. This is the I-won't-know-how-long-it-will-take-until-I've-done-it syndrome. I've had people on courses tell me that their plans were criticized for being "too detailed." I've heard somebody say that too much detail early

in a project is "inefficient."

***This is complete
hogwash***

If there is a Silver Bullet in all of this, some new development, then it is the notion of catching every fragment of detail that you can as early as possible. Only in this way can you build a possible scenario of how the project might turn out.

In reality, you do something slightly more

fancy than producing a single model. You actually produce *multiple* models.

Let me explain. Steps 1-4 cause you to build a prediction of the way the project might turn out. One part of Step 5, Step 5a, gets you to build some contingency, or margin for error into your model. This means that, to some extent, things can turn out differently from your prediction, and it still won't be a problem provided the differences

are covered by your margin for error. Step 5b then allows for the fact that people — your boss, your customer — may not like what your prediction says. This again, is no problem. What you do then is to use your initial model to produce a whole series of models. For example, if they are not happy with the existing delivery date, you can say "OK, here's a model showing what we can do by the date you require,"

or "Here's what we can do if you give us extra people."

This process is described in detail in [Chapters 1](#) through [5](#). Each step is described individually. In [Appendix 1](#), we combine all the steps into a software estimating procedure that you could (a) use, and (b) make part of an ISO 9000 or other process improvement program in the project management procedure.

Having done this much, what we want to do then is to make things actually happen this way, to make reality adhere to the plan, to turn the plan into a self- fulfilling prophecy. That is what Steps 6 – 10 do. And, as you will see, making reality adhere to the plan does not require anything like the god-like levels of ability that such a statement might seem to imply. In fact, you will see that making reality adhere to the plan is one

of the most conceptually simple things imaginable.

Thus you get two chances to make your project a success:

- first, by planning your project in intricate detail;
- then, by causing your plan to become a self-fulfilling prophecy.

Not all of the Ten Steps are equally important.

There is a weighting associated with each step and, collectively, these weightings add up to something we call the Probability of Success Indicator, or PSI. The PSI is an instantaneous measure of how likely or not a project is to succeed. You can read the original derivation of the PSI in [Appendix 3](#). However, I can summarize the weightings as follows:

WEIGHTING

STEP (CONTRIBUTION TO PSI)

1	20
2	20
3	10
4	10
5	10
6	10
7	10
8	10
9	0
10	0
Total	100



In the chapters

describing each of the Ten Steps I will show how that step's individual score is calculated, and how the sum of these scores in turn contributes to the PSI. This will be shown by the "PSI contribution" heading indicated by the icon shown in the margin.

In Chapter 1 of *The House at Pooh Corner* ([Milne, 1923](#)) we come across the following comments by Eeyore, the down-at-heel donkey who inhabits that book:

'It just shows what can be done by taking a little trouble,' said Eeyore. 'Do you see, Pooh? Do you see, Piglet? Brains first and then Hard Work. Look at it! That's the way to build a house,' said Eeyore proudly.

Brains first and then hard work. That's not only the way to build a house: it's the basis for a way to carry out any project.

Structured project management follows this approach in that half of its Ten Steps are to do with planning the project, i.e. they occur before the project really gets moving. This reflects a firm belief of my own: that most projects succeed or fail because of decisions made during this planning phase. Many of the case studies in the pages which follow will serve to illustrate this statement. As for Winnie the Pooh,

we will return to him again in [Chapter 16](#), Coping with stress.

Chapter 1. STEP 1: VISUALIZE THE GOAL; SET YOUR EYES ON THE PRIZE

INTRODUCTION

IDENTIFYING THE GOAL

DEFINING THE GOAL

THE REASON FOR THE
GOAL

MOTIVATING THE TEAM

CHANGES TO THE GOAL
AND CHANGE CONTROL

APPLICATION TO
SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

The first step in structured project management is to identify your goal, your destination. Visualize what the goal is: set your eyes on the prize. There are a number of reasons why you should do this as stated in the following sections.

IDENTIFYING THE GOAL

Step 1 clearly identifies the goal. In the race to the South Pole between Scott and Amundsen ([Huntford, 1993](#)), Amundsen's goal was identified from the outset: he intended to be the first man to stand at the South Pole. Scott's aims were much more diffuse. His expedition was going to be a scientific one; the Pole was a sideshow. His team would do the scientific work; they would also handle the Pole as and

when it came up. Indeed, when Scott discovered that Amundsen was heading south, he disdained any intention of taking part in a race. As project leader of this project, Scott was sending conflicting messages both to himself and to his team. Either the Pole was a goal or it wasn't. If it was, then a project had to be started. If not, then the scientific work was the only project.

DEFINING THE GOAL

Visualizing the goal stating it, writing it down immediately starts two related processes. It begins to tighten the definition of the goal, and it starts the planning process that is the subject of the next chapter.

Tighten the definition of the goal

Amundsen's goal to be the

first man to stand at the South Pole immediately throws up an important point: to bring the team back safely. Ernest Shackleton journeyed to the South Pole in 1909 ([Huntford, 1985](#)) but 97 miles short, with the prize within his reach, he turned back and passed out of the history books. His objective was the same as Amundsen's to be the first man to stand at the South Pole. Ninety-seven miles short he realized that a shortage of provisions would stop him from bringing his team back safely. He turned

back. Thus, for both him and Amundsen, an integral part of the goal was to bring the team back alive.

Visualizing the goal tightens the definition of the basic goal identified in the previous section. Note that for something like the British offensive on the Somme in 1916 — an example I will use later — the definition of the goal may run to many pages.

The definition of the goal is crucially important. What constitutes completion? For

some projects the answer may be as obvious as the nose on your face, but you must consider this issue carefully. If it's so obvious, make a checklist of all the deliverables. To put it another way, write down all the state changes that must occur for the project to be considered complete.

Start the planning process

In visualizing the goal you start to imagine life as it will be

when the project is completed. Mentally you have already made the journey from where you are to this new place. Inevitably, some of the issues that will confront you during this journey start to make themselves apparent.

THE REASON FOR THE GOAL

Setting your eyes on the prize gives you your whole reason for undertaking the project. Life is short and we only pass this way once. If you're going to put some period of your life into a project, then you need to feel that it's going to be worthwhile. Visualization setting your eyes on the prize gives you a glimpse of how you will feel when the project completes.

Visualization involves creating a daydream. In it you start to imagine life as it will be when the project completes. You may dwell on the money you will earn, the recognition of peers and superiors, how your résumé will be enhanced, or perhaps the sense of personal achievement. Whatever it is that drives you, you can get a preview of how you will feel when it is all over. If this preview doesn't fire you up, then for you the project probably isn't worth doing. But if you do feel the adrenalin

starting to flow, then settle into your daydream and allow it to wash around you. Enjoy it. Luxuriate in it. Think of how good you will feel when the project is completed.

MOTIVATING THE TEAM

Visualizing the goal gives you a vision with which to inspire people who may work on the project with you. It's all very well you being fired up, but what about the people who are going to work on the project with you and for projects of any size there will always be other people involved?

Visualizing the goal gives you a vision, a description, of a place somewhere in the future, for

which you are all bound. It draws other people into the daydream. There may be dark days ahead on most projects there generally are and the vision of the goal is what will sustain you and your team during those days. The best example I know of setting your eyes on the prize comes from Martin Luther King's Washington speech in 1963 ([Hampton and Freyer, 1990](#)):

... I say to you today, my friends, so even though we face the difficulties of

today and tomorrow, I still have a dream. It is a dream deeply rooted in the American meaning of its creed, "We hold these truths to be self-evident, that all men are created equal." I have a dream that one day on the red hills of Georgia, sons of former slaves and the sons of former slave owners will be able to sit down together at the table of brotherhood. I have a dream that one day even the state of Mississippi, a

state sweltering with the
heat of injustice,
sweltering with the heat
of oppression, will be
transformed into an oasis
of freedom and justice. I
have a dream that my
four little children will one
day live in a nation where
they will not be judged by
the color of their skin, but
by the content of their
character ...

CHANGES TO THE GOAL AND CHANGE CONTROL

Nobody, least of all myself, is daft enough to believe that the goal, once set, will never and can never change. In the real world we expect that things will have been forgotten, overlooked, appear differently, change or become redundant as the project proceeds. We have no problem with this provided we have a mechanism for controlling these changes.

Any change control mechanism we institute has to operate within the confines of the First Law of Project Management. You already know the First Law; you just may not have seen it formulated in the following way before. The First Law of Project Management states that on any product or system development project there is a function that relates four variables.

These four variables are:

1. Functionality

- Delivery date
- Effort (or cost)
- Quality

The law says that there is a function of these four variables that is constant, i.e.

Function (functionality,
delivery date, effort,
quality) = Constant.

(And if you want to find out more about what this function might be for software

then you could do a lot worse than reading the book *Measures for Excellence* ([Puttnam and Myers, 1992](#)).)

If you change any one of these variables, then all the others will change correspondingly. We have all seen this effect. For example, increase functionality, let's say by adding a new feature, and delivery date will extend or effort will increase or quality will go down, or some combination of these, for the function to remain true.

Much of what is talked about in

this book is opinion (generally mine), or analysis of some event or situation. This, however, is a law a law like, say, gravity. You can certainly pretend that gravity doesn't apply to you. Equally you can pretend this law doesn't apply to you. Both courses of action are about as sensible as each other!

If you believe in the existence of the First Law of Project Management, then it will be your true friend and protector. Pretend it doesn't exist and you will find it the most unforgiving

enemy in the world.

One of the immortal phrases from software projects is "we're absorbing that into the schedule." The First Law of Project Management is the reason why nothing can be absorbed into the schedule. Later on, in Step 5, I will show you a way you can pretend to absorb things, but this is all that it is a pretence. It is a management convenience as much as anything else, a way of not having to go back to your boss or customer to renegotiate the contract every

time a change occurs. But it *is* a pretence, an illusion. The law operates and you had better believe that it does.

Change control can be implemented very simply using a change control log. This can be nothing more than a folder containing a page per change, and a table of contents (with one line per change) which summarizes all the changes. Each change page basically describes:

- the nature of the change

- an analysis of the impact
- what action was taken

Two sample pages, one for changes and one for the table of contents, are presented in [Appendix 5](#).

Ways of visualizing the goal

So how do you go about visualizing and fixing the goal?
Two possible ways are

suggested here, one using a checklist which focuses, among other things, on the day the project will end; the other using a method that I like to think of as the and-they-all-lived-happily-ever-after method.

Visualization checklist

- What will the goal of the project mean to all the people involved in the project when the project completes?

- What are the things the project will actually produce? Where will these things go? What will happen to them? Who will use them? How will they be affected by them?
- What will the completion of the project mean to the team as a whole and to each of its members?
- Why do they want to do this project?
- Why do you want to do it?

- What will life be like on the day/week the project completes?
- What will you do that day? During that week? What will be your routine? Your schedule? Where will you eat? Whom will you meet? What will be the topics of conversation with these people?
- What will people be saying of the project and its deliverables? You? Your boss? The people who

worked on the project? The customer for whom you carried the project out?

- What would you like an audit of the project (see [Section](#) [The reckoning in Chapter 10](#)) to be reporting?
- How will you feel?
- What do you think people will be saying about you? Your boss? Peers? Subordinates? The project's customer? Other parts of

the organization?

- What will be your ambitions/hopes/dreams on that day?
- Will your standard of living have changed?
- Will your position within the organization have changed?
- Will your view of yourself have changed? If so, how?
- Do you think it is a difficult

task you have set yourself?

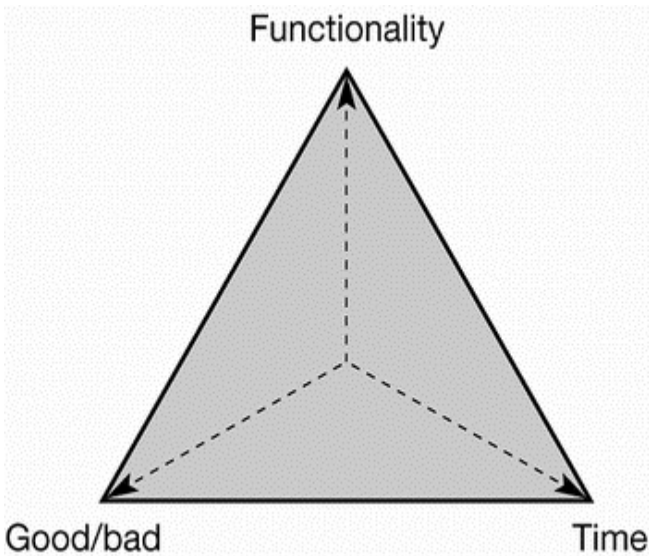
- Could it fail?
- How would you feel then?
What would you do?
- Will you have power you don't have at the moment?
- Will you have changed as a person? If so, how?
- What sort of recognition will you achieve for this project?

- What would you like to do after this project is over?
- What would the best possible outcome of this project be?

The and-they-all-lived-happily-ever-after method

This method involves thinking of the goal of your project in terms of three mutually perpendicular axes as in [Figure 1.1](#).

Figure 1.1.



The first axis is functionality. Low on this axis means minimal functionality, high means all-singing all-dancing. What are you going for here? A lot, a little, or something in the middle?

The second axis is time. When in time will your project end? Will it end when your product is released? Or will it be when your customers have begun to use it and are sending back glowing reports of it? Or will it be when your product has recouped its development costs and is now generating profits

for you? Similarly, for a system development does the project end when the system is released to the users? Or when the users have settled in and become comfortable with the new system? Or when the system has brought the business benefits that were originally set out in the business case? Or when? How you define the end of the project is, to a large extent, arbitrary that is, it's up to you. All that is important is that you fix what lies within the scope of the project and what

lies outside it; that you know what elements form part of your goal and what elements definitely don't.

Finally, a question that seems to be very seldom asked, but is perhaps the most important of all, makes up the third axis. Projects can have lots of different outcomes — bad, average, good, outstanding and the whole gamut in between. What is the best possible outcome from your point of view; the fairytale ending, the one that would really knock everyone's socks off? And just

who are "everyone"? By everyone I mean the team, your boss and the customer. Given that the project can have all these possible outcomes, why opt for an average, scrape-across-the-finishing-line sort of one, when you could have a great one?

Conceptually, the points you select on the three axes fix a point in three-dimensional space which is the goal of your project. Thinking about these three aspects of your project puts a boundary around the goal and determines what lies

within and outside the compass
of your project.

APPLICATION TO SOFTWARE ENGINEERING



As I wrote the first edition of this book, my company was managing a project to develop a large software product for a major telecommunications company. The entire project was estimated to be in excess of 600 man-years, and would last several years. The project

was in a very preliminary requirements-gathering phase.

There was at least one part of the project about 20 man-years of effort which had to be delivered by the end of 1993 at the latest. Let's look at the application of Step 1 of structured project management to that part of the project.

As I have stated earlier, it is important to identify what marks the end of the project. To some extent, this can be an almost arbitrary decision. For example, we might declare the

day the software ships as the end of the project. A slightly better idea might be to declare the ship day plus, say, three months, as the end date. That would give us a chance to see what the customers think of the product. Let's do that then. Let's say that we expect to ship the product at the end of 1993, and three months later, we will declare the project over. By this we mean that we will move the product into a maintenance and enhancement project, and do an audit (see [Chapter 10](#)) as the last act of our project.

Thus, we expect this to be happening about the end of March 1994, all going well. Let's apply Step 1 by following the visualization checklist.

- This project will actually be very significant to all of those involved in it. The organization my company is working with are a very new organization, and this will be their first major project and deliverable. All of the people working on it will be new to the organization, and bringing

it in on time, on budget and producing a quality product will be a tremendous fillip to them. It will also enable the organization with which my company is working, to say to its customer (not to mention its parent) "we can do it, we have justified the faith you had in us."

- We have focused very closely on the things the project will produce. We have done this by visualizing a software catalog in which all of the

individual items which the project will deliver are listed. There is a core software product, three add-on modules, and two sets of documentation.

When the project is finished, these things will be delivered to other project teams who will then build the much larger (600 man-year) system around our deliverables. The deliverables will also eventually go to the end customer, but that is a long time in the future.

- We have already talked about what the project will mean to the members of the project team, and what their motivation might be in doing it.
- The reason I want to do it is that this is the biggest project yet to which structured project management has been applied. I am excited to see how it will work.
- What will life be like on the day/week the project

completes? This question and others like it are ones which enable you to create the daydream we talked about earlier. It is one thing to talk about motivations and other grand ideas, but these are things that can change or drift out of focus with time. On the other hand, thinking about what you will do one day in the future, when a certain set of circumstances have come to pass, is a lot more concrete; and creates a vision which you can lock

onto and carry with you.

- What then will life be like when this project completes? It will be a day at the end of March. The customer will have had the product for three months by then, and we will know from the error reports coming in (or lack of them!) how successful it has been. What we would like would be that there were no error reports at all. (I once had this experience; for six months

no errors were reported.
And yes, the product was in use!)

- To get back to the daydream: We will probably have planned a meeting for that day in March. There will probably have been a regular meeting ever since the product was shipped, to review how the customer was getting on, but this particular day will be the final meeting.
- I will probably get up early,

go for a run and then make my way into the office. I can picture in my head the people who will be at the meeting — people from my company, people from my client's organization and (perhaps) people from the customer, though it is more likely that they will be indicating their acceptance of the product by e-mail, fax or phone.

- We expect there to be a lot of happy people around that day. The client

organization, pleased that their customer is saying nice things about the product and its robustness; the members of the project team, some of them perhaps already assigned to other projects but here for the last act but one of the project (the last act will be the audit); people from my company, pleased that we have delivered on what we set out to do.

- The meeting will be brief, reviewing the errors

reported over the last three months, logging what can be learned from them, spotting where the errors got through our checks, and planning that such things don't happen again. Maybe there will be champagne or cake or some form of little celebration. Maybe there will be bets to be settled (see [Chapter 15](#) on problem solving!). Maybe there will be some nice comments about the product that we can savor

for a while. Maybe there'll be a celebratory lunch.

- OK, I'm sure you get the idea. This is a picture you can enhance and embellish either on paper or in your head, elaborating it with all sorts of incidents, people and words. What is important is to understand what you will feel when the project ends, because from that will stem your motivation. I know what I will feel: just a great sense of achievement, of

happiness for the people who worked so hard to make it happen.

- As regards what an audit will say about it, let's write down the answers we would like to be giving to the audit checklist.
- Did we end up where we said we would? Yes. Exactly. The goal wasn't materially different from the predicted goal. It never changed and we were constantly aware that it hadn't.

- Using the Estimating Score Card in [Chapter 19](#) (or alternatively, reports generated by our project planning tool) we would have captured estimated versus actuals for the project, and we would be able to review these to understand where our estimating needed improvement.
- We would like to be able to say that we did lots right and very little wrong, and here are the lessons we can

learn from the things we did do wrong.

- We would like to be able to say that very few surprises of any significance occurred; and again, of those that did, here are the lessons for the future.
- We would like to be able to say that we didn't have to touch our margin for error; and if we did eat into it in any way, that again there are lessons.

- These are all the things we'd like to be saying in a future audit of our project.
- For my company, this project will be a major feather in its cap, and I will be hoping that we can use it as a springboard to other, larger projects. I'll be hoping that people will be saying that only a company like mine could have done a project like this as well as it was done. I'll be hoping that people will be saying it was a pleasure to work on

the project and that they would like to work with my company again in the future.

- Will my standard of living have changed? Come on, project management doesn't pay that well!
- I don't think my view of myself will have changed, but it's nice for a relatively new company like mine to add something like this to its track record. I think the task we have set ourselves

is difficult enough and has plenty of room for failure. I couldn't bear it if it did fail.

The remaining questions we have probably covered in the course of our ramblings. It's not essential that you answer them all, rather that you get into the thought process behind them. If you end up with a daydream that you can start to carry around in your head, then you can probably press on from Step 1.

PSI CONTRIBUTION



Step 1 contributes 20 towards the PSI. You can think of the score out of 20 as being a measure of how completely well-defined or otherwise the goal of the project is. At the beginning, when the project goal is only a vague idea in people's heads, the score out of 20 will be close to, or at, 0. The score only reaches 20 on the day the project ends. Only then

will you know *exactly* what the goal of the project turned out to be.

The Step 1 score will vary within the range 0 – 20 over the life of the project. Thus, for example, suppose your project life cycle had a number of phases in it, say:

- Requirements gathering
- Design
- Implementation

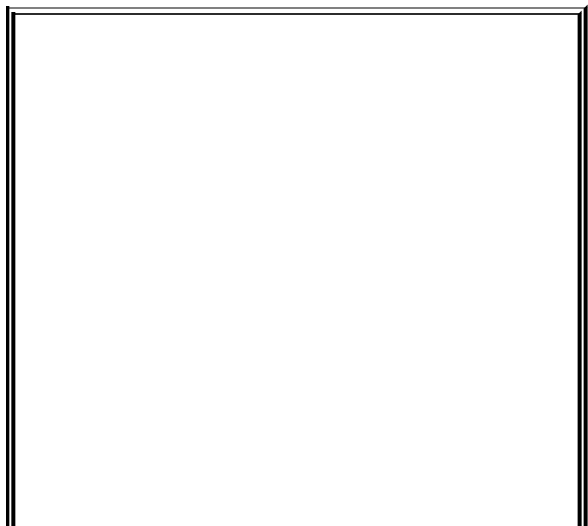
- Integration
- Testing
- Release

then the PSI would gradually increase as you completed each of these phases. If you think of the score as a measure of how well-defined the goal of the project has become as a result of each phase, then you might assign values to these six phases as follows:

Design	9
Implementation	14
Integration	17
Testing	19
Release	20

To use the PSI for your own projects, you should do a similar exercise in which you map the 20 points available onto the phases of your own project life cycle. Remember, the way to do it is to analyze how much more well-defined the goal of the project has become as a result of each phase.

An alternative way would be to break the goal of the project down into blocks and allocate a maximum score per block, the sum of the maximum scores adding up to 20.



STRUCTURED PROJECT MANAGEMENT

Planning the project

- 1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE**

Chapter 2. STEP 2: MAKE A LIST OF THE JOBS TO BE DONE

INTRODUCTION

MAKING A CHECKLIST

IDENTIFYING THE JOBS
WITH FORM 1

APPLICATION TO
SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

I had a boss once, a man from the North of England. No project was too daunting for him, whether it was software development or house renovation. I used to pass the house he was then renovating on my way home from work. Often, in all kinds of weather, his children would be out helping him, shoveling, mixing concrete, carrying things.

"Make a list of jobs that need

doing," was one of his favorite sayings. He would also refer to his children as "the sprogs." In work, we put the two sayings together and came up with, "Make list of jobs and give to sprogs," said in a Yorkshire accent. It seemed the only way to explain how he got so much done in his life.

"Make list of jobs and give to sprogs" make a list of the jobs that need to be done and you've got yourself your project plan. There's only one catch: what if you don't know everything that has to be

done?

Not a problem. At the risk of boring you with references to Polar exploration, a project is just like a journey across featureless wasteland. You know where you want to end up and you know the general direction to take, but you have no idea what lies between here and there; all you can see is the first horizon. What Polar explorers do is steer for a point on that horizon. Only when you get there can you decide how to face the next bit of the journey.

Projects are exactly the same. The very first thing to do when you start a new project is to make a plan. You may only be able to write down the first one or two tasks, but you're already moving forward. Get yourself to that first horizon and then take stock of what needs to be done next. You will always know in quite a lot of detail what lies between you and that first horizon, even though there will inevitably be surprises. For the rest of the project it is enough that you have mapped out in broad terms the remaining

milestones (horizons) along the way.

There will come a time when you have to predict the remainder of the project, but try to postpone that moment for as long as possible, until you have gathered as much information as you can upon which to base your decision.

In the computer industry we call these predictions SWAGs scientific wild-ass guesses but as we shall see, it's not quite as bad as that. Yes, you do have to guess, but by

comparing your early guesses against what actually happened, you can start to get a feel for how accurate your guessing is and adjust your plan accordingly.

Your plan becomes the compass by which you steer the project. As you reach each horizon, you check the lie of the land and then push on to the new horizon. "Make list of jobs and give to sprogs." For us, Step 2 of our method is "Make a list of the jobs that need to be done."

The list can be in any form. It

can be an actual list, like a shopping list; it can be put on to a computerized project planning package we do this for a lot of our software projects; it can be a chart. Many large companies have a defined standard and layout for a project plan, and you may have to follow this. It doesn't matter a toss what it looks like. It doesn't matter that you don't yet know all the jobs that need to be done. Write down the jobs that you know *have* to be done to get the project started and leave the crystal ball-gazing

until you feel yourself better informed.

You will never feel entirely happy about predictions when you are forced to make them. In some situations, for example in business or in the military, you may be forced to make predictions and know that in the first case, maybe your job is on the line, and in the second, lives are on the line. No doubt about it: a lot of projects are about serious things, and a lot of project leaders make decisions which do affect careers and/or lives.

One other thing about jobs: jobs must be explicit. Think through or write down the sequence of events that must happen for the job to be carried out. If you cannot do this, or if you fudge it, then the chances are you're not being explicit enough. Often this is because what you are treating as one job is in fact a series of jobs. What you should do is to break it down further. Remember that jobs can be: (a) serial in nature Job A must be done before Job B can start; (b) parallel in nature Jobs A and B can be

done at the same time; (c)
grouped Job A really consists
of a number of jobs which all
have to complete before Job A
can be considered complete.
(Watch these little guys
they're the ones that will get
you if you're not careful.)

MAKING A CHECKLIST

When drawing up your list of jobs, the following may help to ensure you have covered all possible jobs and areas.

Assuming you have drawn up your basic list, check the following.

- Resources (equipment, products, services, facilities) required.
- Skills required and whether

these imply hiring and/or training.

- That you have listed explicit, clearly identifiable milestones.
- Timescales, costs and budgets show how you arrived at your estimates.
- That you have explicitly stated what assumptions you are making.
- That you have explicitly stated what dependencies

there are on things which are not directly under your control.

- That you show explicitly who is responsible for what.
- That you have given some thought to what the really high-risk areas are.

IDENTIFYING THE JOBS WITH FORM 1

The form shown in [Figure 2.1](#) can be used to help identify all the jobs in a project. The project is written in the box at the top and then the jobs that go to make up the project are written sequentially underneath. Because of our recursive definition of *project*, any job can then be further broken down by writing its name in the box at the top.

Figure 2.1. Form 1, job list

Form 1

Project

Jobs _____

Jobs _____

Jobs _____

Jobs _____

Jobs _____

Jobs _____

Jobs _____

Jobs _____

APPLICATION TO SOFTWARE ENGINEERING



As soon as a project gives any sign at all of starting, we are in a position to make a list of jobs for it. As we have already discussed, the list should be detailed to the first milestone and give the major milestones thereafter. In addition, what you can do is to add any other

available detail to the remaining milestones. This isn't essential, but given that we may have such detail available, and that we will have to put it in some day, we may as well do it now. The advantage of this is that it starts to give us some feel for the magnitude of the remainder of the project. (We will return to this idea in [Appendix 1.](#))

In the two sections that follow I have given real-life examples of job lists. The project for Telecom Ireland Software (TIS) has four major jobs within it

(remember our recursive definition of project):

- Decide to implement ISO 9001
- Set up organization
- Set up and install quality system
- Apply for certification

The first job has been further broken down into four jobs, and these have been estimated at a total of five man-days. The

remainder of the job list then gives as much as is known about the rest of the project. Note that I have gone down, wherever possible, to very low levels of detail. This will be the basis of our estimating method in [Appendix 1](#).

EXAMPLE 1: PRELIMINARY PLAN CERTIFICATION

This example is a project to achieve the ISO accreditation ([Rothery, 1991](#)).

1. Decide to implement ISO 9001

Develop a quality policy statement and have it signed by the top management, TIS and explained to everyone in ACME Software. The cost of the project is estimated. Includes:

- Develop statement
- Review internally
- Take to board for approval
- Circulate to everybody
- Education, i.e. Standards Bodies (videotapes, manuals, etc.) give an introductory course, ongoing

of quality program, issues and results

2. Set up organization

- Nominate a quality manager
- Set up a quality improvement team

3. Set up and install quality system

This involves installing and running a quality manual. There are two sides to the describing the quality system ("document v the system described in the quality manual

These two aspects of the work would typically framework for a quality manual has already work involved in achieving ISO 9001 is to c implement it and document it within the qu

The bulk of the rest of this document estimates terms of setting up/installing the procedure manual. Some notes on implementing the

- it would be useful to review the content Standards Bodies or a quality auditor/
- plan should allow for up to ten man-d
- integral elements in implementing the
 - the measurement of the quality which produce them
 - calculating the cost of quality
 - taking corrective action based o

Note

In the following example all estimates

Introduction

- Company
description
- 1.1 extract from
marketing
brochures
- Place of quality
manual in
- 1.2 overall
procedures -
written
- Quality policy
statement (get
- 1.3 from 1 above)
and education
program

- 1.4 Use TIS Organization chart
- 1.5 Define an amendment procedure
- 1.6 Circulation list and procedure

Totals

Documentation

- 2.1 Use Myles' document
- 2.2 Use Myles' document Document

2 management
(2 documents)

- 2.3 - Write
- Review
 - Update
 - Approve
 - Publish

Totals

Project management

Project
management
methodology
(adopt?

3.1 Document w.r.t
book. Project
exit criteria

exit criteria,
sample project
plan)

Estimating &
scheduling
(document w.r.t
book & MS

3.2 Project. Include
examples;
historical
database)

Monitoring &
reporting
(needs an EV
system; project
status report;
sample

progress
report)
Include
timesheets and
time recording

Totals

Project Life Cycle

4.1 Overview

Requirements
spec.

- Research
- Write
(including

4.2 template)

- Review

(possibly by
using)

- Rework
- Release

High-level
design

4.3

- Research
- Write (inc.
deliverables,
review/exit
criteria,
template)
- Review
(ideally by
using)
- Rework

- Release

Low-level
design (inc.
user
documentation
and test plans)

- Research

4.4 - Write (inc.
template)

- Review
(ideally by
using)

- Rework

- Release

Implementation

(inc. code unit

(inc. code, unit
test,
integration and
test)

- Research (inc.
unit
development
folder (UDFs),
walkthroughs,
etc.)

4.5

- Write (inc.
deliverables,
review/exit
criteria,
template)

- Review
(ideally by

using)

- Rework

- Release

Alpha test

- Research

- Write (inc. deliverables, review/exit criteria,

4.6 template)

- Review (ideally by using)

- Rework

- Release

Beta test

Beta test

- Research
- Write (inc. deliverables, review/exit criteria, template)

4.7

- Review (ideally by using)

- Rework
- Release

Operation and maintenance

- Research
- Write (inc.

4.8 deliverables,
review/exit
criteria,
template)

- Review
(ideally by
using)

- Rework

- Release

Configuration
management
and change
control

- Research

- Write (inc.

- 4.9 deliverables,
review/exit
criteria,
template)
- Review
(ideally by
using)
 - Rework
 - Release

Totals

Invoicing and accounting

5 Guess

Training

6 Will refer to TIS
training plan (to be

estimated
separately)

7 Subcontractors

Guess

Orders and enquiries

Not required by ISO

8 9001 but a good
thing to include.
Includes prospects
list.

Guess

Personnel

9.1 General
procedures

Organization

9 9.2 chart - done

Grading

9.3 structure/Job
descriptions

9.4 Salary scales

Totals

The quality system Itself

Internal audits

-

10.1 review/update
of procedures
(quality)

10

Cost of quality

10.2 - maintenance
of quality

records (guess)

Totals

4. Apply for certification

Sequence of events is:

Run an internal audit (or perhaps them)

Submit application form and a quality manual (QM)

Standards Bodies vet QM and note discrepancies which must then

When these are rectified, Standards audit TIS onsite and again identify discrepancies

Standards Bodies verify remedial action taken and recommend award

certificate

Total

EXAMPLE 2: PLAN FOR A SOFTWARE ENGINEERING PROJECT

The second example is for a software engineering project as shown in [Figure 2.2](#). This job list was done using MS Project.

Figure 2.2. Example of project plan

			2000											
ID	Task Name	Work	D	J	F	M	A	M	J	J	A	S		
1	1 The Project	503.5 days												
2	1.1 START	0 days												
3	1.2 Project planning and scoping meeting	9 days												
4	1.3 Produce Requirements Document	27 days												
5	1.3.1 Research user requirements	7 days												
6	1.3.1.1 Gather info on competitive products	0.5 days												
7	1.3.1.2 Review with marketing	2 days												
8	1.3.1.3 Identify users	0.5 days												
9	1.3.1.4 Prepare user questionnaires	2 days												
10	1.3.1.5 Distribute questionnaires	0.5 days												
11	1.3.1.6 Retrieve questionnaires	0.5 days												
12	1.3.1.7 Analyse information	1 day												
13	1.3.2 Write requirements document	9 days												
14	1.3.3 Review cycle	10.5 days												
15	1.3.3.1 Circulate	0.5 days												
16	1.3.3.2 Individual review	2.5 days												
17	1.3.3.3 Review meeting	3 days												
18	1.3.3.4 Changes to document	2.5 days												
19	1.3.3.5 Circulate again	0.5 days												
20	1.3.3.6 Second review	1.5 days												
21	1.3.4 Signoff	0.5 days												
22	1.3.5 Requirements complete	0 days												
23	1.4 Produce System/Acceptance Test Plan	20.5 days												
24	1.4.1 Research	5 days												
25	1.4.2 Write Navigation tests	3 days												
26	1.4.2.1 Define test sequence	1 day												
27	1.4.2.2 Write test scripts	1 day												
28	1.4.2.3 Define expected results	1 day												
29	1.4.3 Write Functionality tests	3 days												
30	1.4.3.1 Define test sequence	1 day												
31	1.4.3.2 Write test scripts	1 day												
32	1.4.3.3 Define expected results	1 day												
33	1.4.4 Write Integrity tests	3 days												
34	1.4.4.1 Define test sequence	1 day												
35	1.4.4.2 Write test scripts	1 day												
36	1.4.4.3 Define expected results	1 day												

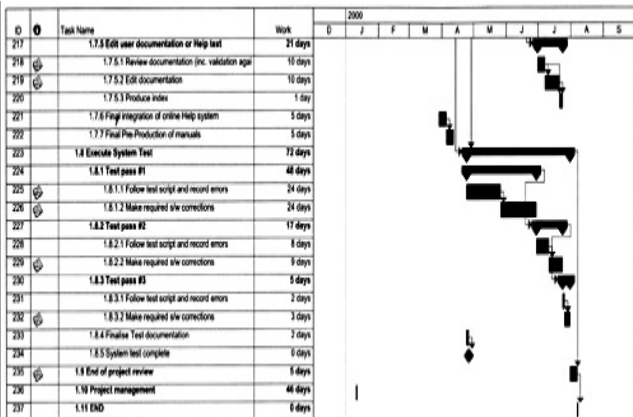
ID	Task Name	Work
37	1.4.5 Write Performance tests	3 days
38	1.4.5.1 Define test sequence	1 day
39	1.4.5.2 Write test scripts	1 day
40	1.4.5.3 Define expected results	1 day
41	1.4.6 Write Stress tests	3 days
42	1.4.6.1 Define test sequence	1 day
43	1.4.6.2 Write test scripts	1 day
44	1.4.6.3 Define expected results	1 day
45	1.4.7 Write Installation tests	3 days
46	1.4.7.1 Define test sequence	1 day
47	1.4.7.2 Write test scripts	1 day
48	1.4.7.3 Define expected results	1 day
49	1.4.8 Review cycle	16 days
50	1.4.8.1 Circulate	0.5 days
51	1.4.8.2 Individual review	5 days
52	1.4.8.3 Review meeting	3.5 days
53	1.4.8.4 Changes to document	5 days
54	1.4.8.5 Circulate again	0.5 days
55	1.4.8.6 Second review	1.5 days
56	1.4.9 Signoff	0.5 days
57	1.4.10 System Acceptance tests written	0 days
58	1.5 Prototype	22 days
59	1.6 Write code	184 days
60	1.6.1 Pricing	8 days
61	1.6.1.1 Write code element (including clean comp)	4 days
62	1.6.1.2 Document code element	0.5 days
63	1.6.1.3 Unit test code	3.5 days
64	1.6.1.3.1 Prepare test Plan and test set	1 day
65	1.6.1.3.1.1 Define test sequence	0.25 days
66	1.6.1.3.1.2 Write test scripts	0.5 days
67	1.6.1.3.1.3 Define expected results	0.25 days
68	1.6.1.3.2 Subsystem ready for unit testing	0 days
69	1.6.1.3.3 Test code	1 day
70	1.6.1.3.4 Make corrections to code	0.5 days
71	1.6.1.3.5 Test code again	0.5 days
72	1.6.1.3.6 Prepare test report	0.5 days

[illegible]

graphics/02fig02d.gif

graphics/02fig02e.gif

ID	Task Name	Work
181	1.6.9.1 Define test sequence	1 day
182	1.6.9.1.2 Write test scripts	2 days
183	1.6.9.1.3 Define expected results	1 day
184	1.6.9.2 Subsystem ready for unit testing	0 days
185	1.6.9.3 Test code	6 days
186	1.6.9.4 Make corrections to code	2 days
187	1.6.9.5 Test code again	2 days
188	1.6.9.6 Prepare test report	2 days
189	1.6.10 Online help	32 days
190	1.6.10.1 Write code element 1 (including clean co	3 days
191	1.6.10.2 Write code element 2 (including clean co	3 days
192	1.6.10.3 Write code element 3 (including clean co	3 days
193	1.6.10.4 Write code element 4 (including clean co	3 days
194	1.6.10.5 Document code element 1	1 day
195	1.6.10.6 Document code element 2	1 day
196	1.6.10.7 Document code element 3	1 day
197	1.6.10.8 Document code element 4	1 day
198	1.6.10.9 Unit test code	16 days
199	1.6.10.9.1 Prepare test Plan and test set	4 days
200	1.6.10.9.1.1 Define test sequence	1 day
201	1.6.10.9.1.2 Write test scripts	2 days
202	1.6.10.9.1.3 Define expected results	1 day
203	1.6.10.9.2 Subsystem ready for unit testing	0 days
204	1.6.10.9.3 Test code	6 days
205	1.6.10.9.4 Make corrections to code	2 days
206	1.6.10.9.5 Test code again	2 days
207	1.6.10.9.6 Prepare test report	2 days
208	1.7 Produce User Documentation and Help system	99 days
209	1.7.1 Research requirements	5 days
210	1.7.2 Set up environment and tools	5 days
211	1.7.3 Define style sheet and produce prototype	5 days
212	1.7.4 Write user documentation or Help text	53 days
213	1.7.4.1 Structure into chapters	1 day
214	1.7.4.2 Produce Table of Contents	1 day
215	1.7.4.3 Review Table of Contents	1 day
216	1.7.4.4 Write documentation	50 days



Don't spend too much time going into these figures in any detail (for one thing, they're not complete, so don't assume that, for example, you'll get ISO 9001 accreditation by following them!). They are merely snapshots of projects at a particular point in time. They are given here to illustrate the following important points in connection with making job lists.

- Use a project planning tool, such as MS Project, if at all possible they give so much more clarity as I think is illustrated by the difference between the plans in [Example 1](#) and [Example 2](#). The effort involved in learning and inputting information to them will be repaid a hundredfold in the understanding, flexibility, responsiveness and breadth of vision that this computer model of your project will give you. As I become older and more set in my ways, I venture to suggest that anyone who doesn't use one of these things is nuts!
- Write down the major milestones.
- Fill out the first milestone in complete detail.

- Fill out the remaining milestones with as much detail as is available (you can use the checklist given previously to assist you in doing this).

And what level of detail are we talking about? This is one that always seems to cause people problems. Well, note that the example has 500 man-days in it. The job list contains 237 jobs. This means that on average each job covers between two and three man-days. More generally, I think somewhere between one and five man-days is an acceptable level of detail to aim for. A pain in the neck? You bet. But this is the only way that success can be assured by ensuring that no matter how big the project is, that each component is planned and tracked by someone at the day level of detail. It doesn't have to be you, and on a big project it won't be but someone needs to do it.

I know the following to be absolutely true: when you break it right down, any project ends up as a large number of occurrences of Joe Soap or Jill Bloggs spending two days here, or half a day there doing Job A prior to Job B. Now, as the project manager, there are two crucial issues that you have to address. These are: (1) are we consciously going to decide who will work on what when? and (2) if so, when are we going to decide?

The first is not as dumb as it sounds. All project managers hand out work assignments which are more or less detailed. There is, then, an inherent assumption that these assignments will get translated into the two days here/half day there we talked about earlier. If they do, fine. But if they don't then it means that some force other than the project manager let's call it Life is making these work allocations. The allocations will get done, have no fear about that,

because if you don't do them, Life will. But one of the things about Life is that it can be unfair, and if you rely on Life to do your work allocations for you, then you're asking for trouble.

Let's look at the second issue. Assuming you're consciously going to allocate your people to jobs, when should you do this? Well, you have a choice. You can either do it at the beginning, during the planning part of a project when things are relatively quiet, or you can do it in the middle of the project, in the heat of battle, when the bombs are falling and all hell is breaking loose. You *will have* to do it, so why not make things easy on yourself, and do as much as you can at the beginning when the pressures aren't so bad? And don't worry about it not getting done because, as before, Life will intervene and do it for you. But again, you need that kind of assistance like you need a hole in the head.

Let's try to summarize what we're saying here about levels of detail. There is a quote in probably the most famous book on software project management, *The Mythical Man-Month* ([Brooks, 1975](#)): "*How does a project get to be a year late?*" The answer? "*... one day at a time.*"

Know where your man-days are going, every one of them. Spend them wisely. If you don't, you'll find out just how fickle Life can be.

Once, at a seminar, I was pushing the idea of the plan being at the day level of detail, and somebody asked whether this didn't present an unacceptable overhead. Specifically, the question was whether I had estimated the effort/cost involved in the management of Scott's project versus that of Amundsen. There are two answers to that question. The first is that if we assume that Scott did all the project management on his project and

Amundsen on his, then both put in almost exactly the same amount of effort three years. The second answer is this, and it is not intended to sound clever. Amundsen had a successful project. Thus whatever the management of it cost, it was worth it. Scott's project cost the lives of five people. At this kind of price there is no such thing as an *unacceptable overhead*.

PSI CONTRIBUTION



Step 2 contributes 20 towards the PSI. You can think of the score out of 20 as being a measure of how completely well-defined or otherwise the plan for the project is. At the beginning, when the project is just starting, the plan probably contains nothing more than the major milestones with some, probably incomplete, detail thrown in. Then the score out

of 20 will be very low. On the day the project ends the plan will be complete, and will show the myriad little jobs, all strung together, that went into carrying out the project. This is the only time the score can reach 20.

So, between the beginning and end of the project, the score will vary within the range 0 - 20. Thus, suppose your project life cycle had a number of phases in it, say:

- Requirements gathering

- Design
- Implementation
- Integration
- Testing
- Release

The PSI would gradually increase as you completed each of these phases. If you think of the score as a measure of how much closer you are to knowing all of the jobs that have to be done to complete the project,

then you might assign values to these six phases as follows:

Requirements gathering	4
Design	9
Implementation	14
Integration	17
Testing	19
Release	20

To use the PSI for your own projects, you should do a similar exercise in which you map the 20 points available onto the phases of your own project life cycle. Remember,

the score is a measure of how close you are to knowing all of the jobs that have to be done to complete the project.

As before, an alternative way would be to break the overall project plan down into sub-plans for each of its individual components. Then, allocate a maximum score per sub-plan, the sum of the maximum scores adding up to 20.



STRUCTURED PROJECT MANAGEMENT

Planning the project

- 1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE**
- MAKE A LIST OF THE JOBS THAT NEED TO BE DONE**

Chapter 3. STEP 3: THERE MUST BE ONE LEADER

INTRODUCTION

A ROLE MODEL

APPLICATION TO
SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

"But of course there is a leader," I hear you say, "it's the first thing we do when we start a new project. We appoint a leader."

And indeed this may well be true. Whatever else may be said about most projects that we come across, we can pretty much always identify the leader. And isn't it quite a nice sounding title to give to somebody or to have

ourselves? "Project Leader."
"Project Manager." It's a title
that settles easily on the
shoulders like a fine coat. We
walk that bit taller with the
responsibility and pressure that
we now carry. Yes indeed, we
can always point out the
project leader if nothing else
we can tell him by the way he
walks!

Let me re-state the chapter
title. The project must have
one leader. By that I mean it
can't have no leaders and it
can't have two leaders or a
committee of leaders. *There*

must be one leader.

Let's look at this statement a bit more closely. When I say *leader*, I mean not so much the person with the title, but a person who is going to get the project done. She lives, eats and breathes the project. She is going to get it done or die in the attempt. At any given time she has her finger on the pulse of the project and knows how it's proceeding. Her name is so intimately bound up with the project that when you think of one you think of the other. A fairly negative way of looking

at it is that she is the person whose ass is pinned to the project and who will get fired if it doesn't work out. In the terminology of other books she is the project champion.

I have worked on a project where there was no leader, in the sense that I have described it above. I have worked on a project where there was a formal leader, and a person who took over the psychological leadership, thereby resulting in two leaders. Both projects were unmitigated disasters. In the

first case, the person with the title project leader was fired, and in the second, both the project leaders were fired.

As you can see from the case studies below, sometimes the problem is finding a person, sometimes there are too many contenders.



CASE STUDY 1

One thing that can happen to you as a project leader is that your leadership can be subject to challenge by another member of the team. This means that somebody else tries to take over the psychological leadership of the project from you. If this happens you must neutralize the challenge. If you don't, you will end up with multiple leaders on your project.

Amundsen the project leader had a leadership challenge during his expedition to the South Pole in 1911 ([Amundsen, 1912](#); [Huntford, 1993](#)). One of his men, Hjalmar Johansen, was a Polar explorer in his own right. He had been on an earlier expedition with Fridtjof Nansen, the father of Norwegian Polar exploration, and

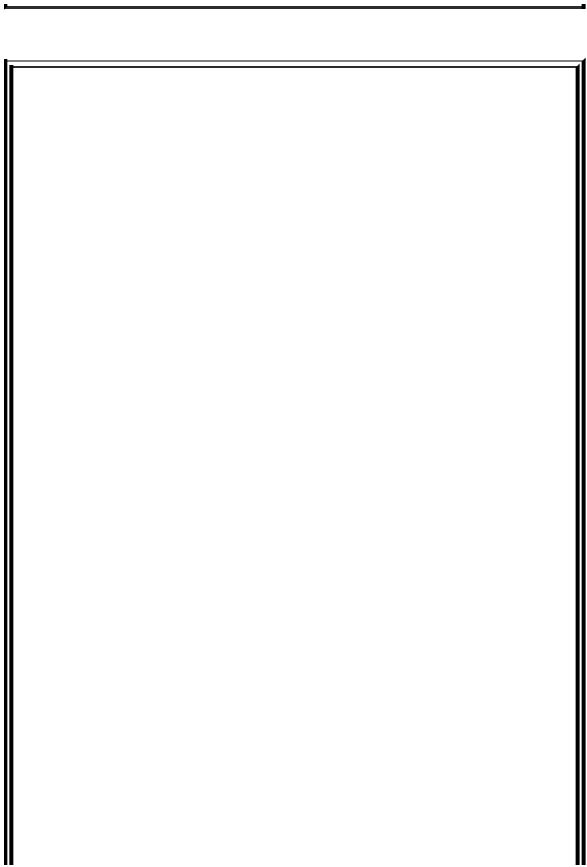
together they had made a dash for the North Pole. They didn't reach it but they did get 170 miles closer than anybody else had done, reaching $86^{\circ} 14'$ North. Amundsen had reluctantly brought Johansen, urged to do so by Nansen.

From the beginning of the voyage, Johansen began to compare Amundsen's voyage unfavorably with Nansen's. Johansen was a better skier and dog-driver than Amundsen and felt he knew more about Polar travel. Often he would correct or contradict Amundsen, or volunteer advice. These matters later came to a head when Amundsen set out on his Polar journey. Amundsen was anxious to set out for the Pole, plagued by the thought that Scott might already have started and that Scott had brought motorized transport with him. His colleagues,

and particularly Johansen, warned him against too early a start.

Amundsen wanted to start on August 24th but having twice put it to a vote and having been defeated he yielded. Finally, on September 8th, 1911, which is still the Polar winter, Amundsen could restrain himself no longer, and the party of men, sledges and dogs set off from their base.

The start turned out to be a disaster. Temperatures plummeted; dogs died from the cold; liquid froze in compasses; and the men eventually turned back. At breakfast, after they had all returned, Johansen took Amundsen to task for what had happened. This challenge to his leadership was too much for Amundsen. Johansen was removed from the party going to the Pole. Years later he would commit suicide.

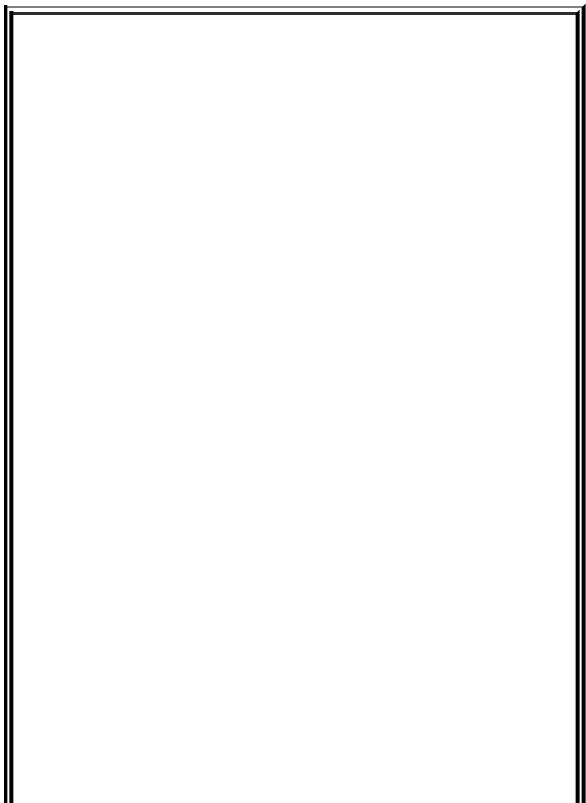


CASE STUDY 2

I have stated that not having one leader will guarantee failure.

Unhappily, the converse isn't true: having a leader doesn't guarantee success. Scott ([Huntford, 1993](#)) is a case in point. He was all the things I have indicated above. He did live, eat and breathe the project. He was going to get it done or die in the attempt. Scott reached the South Pole a month after Amundsen and perished with all four of his team on the return journey.

In terms of structured project management, it was Scott's failure to carry out Steps 2, 5, 6 and 8 which sealed his fate. To reiterate, there must be one leader. Not zero, not two and not a committee. I believe your project will fail if this is not so.



CASE STUDY 3

I remember my first day on the job in a new company. I was given a copy of the Project Status Book. As the name suggested the book contained the state of every project in the organization. It was updated weekly, well laid-out and very pretty. Each page had some summary information about the project at the top; name, start and end dates, project leader, and so on.

I was amazed to see the same handful of names in all the project leader slots. There were people in there handling ten, fifteen projects. I was young then, and very impressed; I thought these guys must be absolute tigers to be able to stay on top of so many things. It was only later I discovered that they

weren't on top of all these things, and that their names up there in the heading meant nothing. Sure they were responsible, maybe their necks were even on the line for those projects, but they sure as hell weren't leading them. There must be one leader.

CASE STUDY 4

One idea I have come across a few times in software projects is that of having a technical leader and a managerial or administrative leader. The managerial leader is more a manager-type person who has no interest in technical issues, while the technical leader doesn't want to be a manager, but still wants to have a big say in what goes on.

Forget it. Unless there is one manager and he has the final, overall say, then a Laurel and Hardy approach like this won't work. There must be one leader.

A ROLE MODEL

If we are looking for a project manager to act as a role model, we can do no better than to look at some of those old cattle-drive Westerns with John Wayne as the trail boss. (You could also read *Lonesome Dove* ([McMurtry, 1990](#)).) I'm sure we're all familiar with the scenario: several thousand head of cattle have to be driven from the Rio Grande to a railhead, usually either Kansas City or Abilene, and the entire

film then consists of all the incidents that occur on the way.

If we compare the jobs on your job list to the cattle in the cattle drive, then we have a very good analogy for how projects actually proceed. At any given time in the cattle drive, we can probably identify a steady state situation something along the following lines.

Most of the herd are heading generally in the direction of Abilene. (This excludes the mandatory stampede sequence

where the entire herd heads off to some place entirely different. Hopefully you will never have a stampede on one of your projects!)

While the majority of the herd are behaving themselves, it is however probably safe to assume that some of the herd are heading backwards to the Rio Grande they have no plans to go to Abilene or any place the trail boss (John Wayne) has in mind for them. Others have drifted off to a gulch (ravine) where they've found some sweet grass and

are grazing contentedly. Others are at a watering hole and are keeping cool, and haven't the slightest interest in getting back on the march under a hot sun. Finally, some cattle have perhaps wandered on ahead and been rustled.

John Wayne, meanwhile, spends the entire film riding around the herd, bringing up the stragglers, scouting on ahead for trouble, and generally making sure that all the cattle are headed for Abilene.

Without wishing to labor the analogy any more, this is exactly what the project leader has to do to make sure that all the jobs are being pushed forward, and that the entire *herd* of jobs is moving in the right direction.

Note, too, that if you decide to be this kind of project leader then this may well generate more jobs for your job list. The logic is simple and unfaultable: if you want a successful project, then any job that is required for the success of the project is your responsibility.

Whether these jobs are formally your responsibility or not; whether the people doing them lie directly under your control or are peers, superiors or members of different organizations entirely; irrespective of any of these considerations, the jobs involved become part of your *herd* and you have to make sure they happen. I'm not saying you *do* them. Not at all. What I am saying, though, is that you have to cause them to happen. And not doing so will be no consolation. If the project

ends unsuccessfully, then it'll be small consolation to be able to say "well, that wasn't my job."

To repeat the point once more so as to make it crystal clear and unambiguous: if you want a successful project, then any job required for that success is the project leader's responsibility, and she has to take it under her wing and take it forward. *This* is what being the one leader means.

APPLICATION TO SOFTWARE ENGINEERING

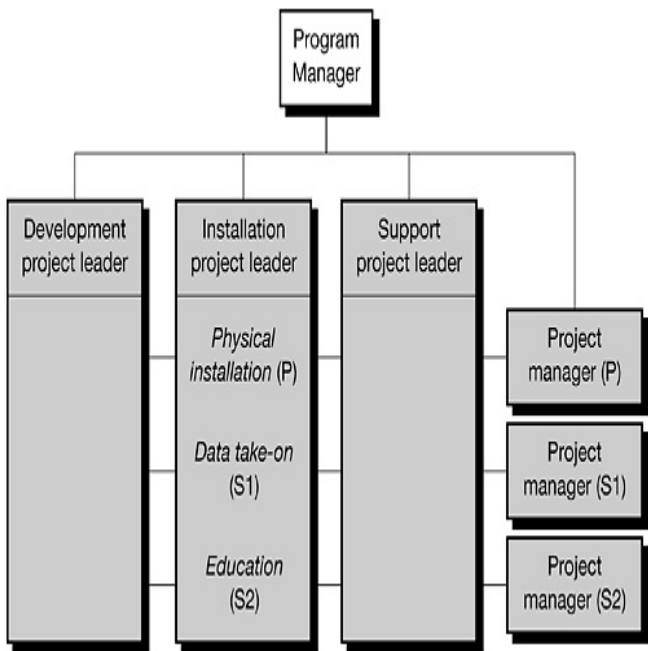


Here are a couple more examples just to hammer home the point about the need for a single leader. In the software engineering project we've been looking at in [Chapters 1](#) and [2](#), what is the story with its leader?

Well, the setup is that we have a small team of seven people, most full-time but a couple part-time, and one of the full-time people is the leader. Nice. Couldn't be simpler. Now let's look at a more complex example.

[Figure 3.1](#) shows a proposed organization chart for a large project involving a consortium of three companies — the prime contractor, P, and two subcontractors, S1 and S2.

Figure 3.1.



There are three logical units in the project:

- development
- installation
- support

Within each of these units, each of the contractors will have certain roles to play. For example, in the installation unit, let's say for example that Contractor P might be

responsible for physical installation of the system, S1 might be responsible for data take-on and S2 responsible for education.

Each logical unit will have a project leader. In addition, each contractor will have a project manager responsible for its components across all three units. What, if anything, is wrong with this arrangement? The test is simple. Each project must have a leader, so let's see how it shapes up.

The project as a whole has a

leader the box marked program manager. Each unit also has a leader, the project leaders. Oh, so what are the project managers doing?

Nothing because the project leaders are doing it?

Duplicating what the project leaders are doing? I don't know myself. Clearly, there are two ways to settle this. Either (1) have three people, one responsible for each of the logical units, or (2) have three people, one responsible for each contractor's contribution. All the way down the tree, a

single person must be identified whose ass is on the line for each particular part of the project. This is what Step 3 is all about.

Thus, in our example, at the next level down, there would be a person responsible for each of the components such as installation, data take-on, education, etc. Be careful of organization charts. They can give the illusion that there is a single leader at each level in the hierarchy. However, look behind them, look at the one for your own organization. For

example, see if the one leader principle applies if there really is a person whose ass is on the line for each part of the project that makes up your organization.

PSI CONTRIBUTION



To get most or all of your ten points here is very simple. The ten points is a measure of how much you are going to embrace all the jobs on a project and take them forward. Pause for a moment before you award yourself an automatic ten here.

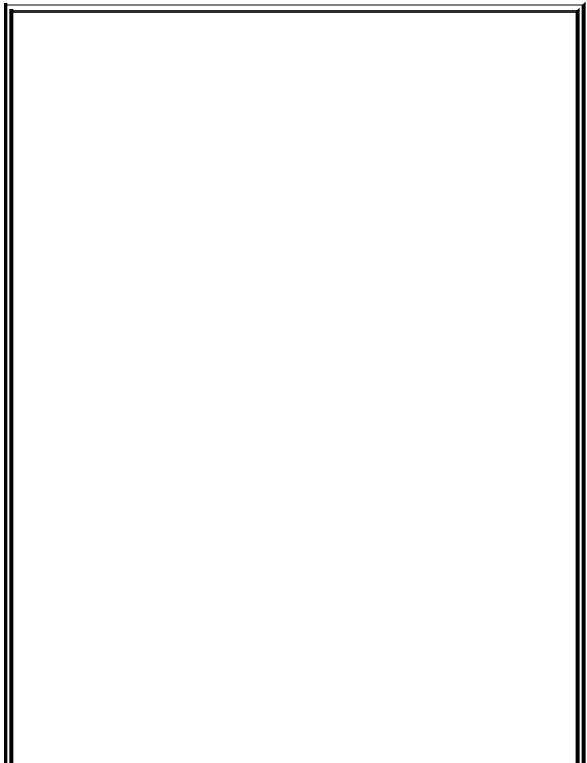
A ten means that you are going to move heaven and earth, do whatever has to be done to

make a project happen.
Irrespective of formal
structures, political
considerations, resource or
personnel constraints, famine,
fire, flood, pestilence, war,
nothing is going to stand in
your way.

Now it seems to me that as
human beings we may be able
to bring such energy to bear on
some projects. By and large, I
don't think we can do it on all
projects we become involved
in. This, of course, merely
reflects the rather obvious fact
that some projects are more

important than others. Thus, there may be some projects under our control that we are going to do everything possible to make happen. Then, fair play, we draw our ten points. Similarly, though, there may well be others where we decide that we just aren't prepared to give them that level of energy. There is no inherent problem with this provided we recognize that the lower score we award ourselves here reflects the fact that things may get forgotten or fall between the cracks, and we need to perhaps keep a

weather eye out for these.



STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- **MAKE A LIST OF THE JOBS THAT NEED TO BE DONE**
- **THERE MUST BE ONE LEADER**

Chapter 4. STEP 4: ASSIGN PEOPLE TO JOBS

INTRODUCTION

EACH JOB HAS A NAME

PEOPLE'S OTHER
COMMITMENTS

MONOLITHIC TEAM OR
FLAT STRUCTURE

HIERARCHY OR TEAM
STRUCTURE

MAXIMIZE STRENGTHS

ASSIGNING PEOPLE TO
JOBS

ASSIGNING PEOPLE TO
JOBS WITH FORM 2

APPLICATION TO
SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

Many of the projects we encounter are too big for one person, and are carried out through using other people. When I first drafted this section of the book, the Gulf War was raging. This was a project which had nearly three quarters of a million people on the Allied side alone.

The overall goal of the project can be thought of as a picture broken down into pieces like a

jigsaw puzzle. There are as many pieces as there are people. If each person completes his piece, the picture will complete, and the goal will be achieved. (Note that by our definition, each person becomes engaged in his own individual project and can apply structured project management to that project.)

There are three things you need to do as part of this step:

- 1. Make sure each job has a name against it**

- Take people's other commitments into account
- Try to maximize the strengths of the team you've got.

We discuss each of these in turn.

EACH JOB HAS A NAME

One of the things we do in my company is to turn around projects that are running out of control. There are basically two phases in doing this. One is to understand why the project is the way it is; the other is to do a new plan (that is, apply Steps 1 – 5) to take the project from where it is now to where it needs to be. One of the first things we do in trying to understand why the project got into a mess in the first place is

to take a look at the current project plan.

Sometimes there isn't one! If that's the case then the investigation is pretty much over.

More often though there is a plan of sorts. That being the case, one of the most common things we find is that jobs don't have people's names against them. What you find instead are things like the following:

- ANO (also known as "A.N.

Other")

- Mr (or Ms) X
- Programmers 1 6
- S/W Eng.

and other equally mysterious personages. In other words, the jobs were never assigned to anybody. You might think that people wouldn't be too surprised then if these jobs weren't done. But, invariably, surprised is exactly what they are.

The first thing in assigning people to jobs is that every job must have a human being's name against it. Any of the above or organization names should be avoided as far as is humanly possible. Even if you are subcontracting something so that an organization is actually working on one of the jobs on your project, the name you should have against that job is the member of that organization whose ass is on the line for the successful delivery of that particular job.

It may be that at the beginning

of a project you don't know who all the people will be, and that's OK, but then one of your priorities as project manager has to be that the unknowns get replaced by warm, living, loving human beings as quickly as possible.

PEOPLE'S OTHER COMMITMENTS

We once turned around a software development project that was meant to last a year and ended up lasting nearly four years. When we looked back over its history to try to understand what had happened, here's what we discovered.

The original estimates of effort were out by about 50 percent, which in my experience of

software projects is not too bad at all. Quite creditable actually. But then, we discovered that the project manager had made a kind of unconscious assumption that everyone would be available to the project full-time. When we checked back over timesheets and other data, we discovered that people had actually worked on the project on average between 1.5 and 2 days per week. Now, I have a degree in applied maths, but you don't need a degree in anything to figure out that a

project like this is probably going to run about three years late, which is exactly what happened. Needless to add, the original assumption about staffing levels unconscious or otherwise was lost along the way, and all that everyone saw was a project that was years behind schedule.

The point here is that we have to take people's other commitments into account. PC-based project planning tools have facilities for doing this where you can allocate people to projects a certain percentage

of the time, but at its most simple, all you have to do is make a list for each person that looks, for example, like this:

Reggie

- | | |
|---|-----------------------------|
| • The other project | 2 days per week |
| • Personal development | 2 hours (0.25 day) per week |
| • ISO 9001/Quality | 0.5 days per week |
| • <i>Therefore, available to my project</i> | <i>2.25 days per week.</i> |

Note, in particular, that annual leave, public holidays and sickness are all commitments that have to be allowed for.

This is the supply side of a supply/demand equation where the demand has been generated by the list of jobs and associated dependencies and effort derived from Step 2. Step 2 identifies the pile that has to be moved; Step 4 shows how it will be moved.

Note that this business also applies to yourself as project leader. Often, especially in

software projects, project management is what the project leader does when he has any time left over from doing technical work. A rule of thumb that we offer is that you should reckon on 6–8 percent of total project effort for project management: 6 percent would apply to smaller projects—say, 6 people or less—while 8 percent would be for larger ones—say, above 12 people.

This figure is intended to cover work by any member of the project team on:

- project planning activities including scheduling, estimating and updating of PC-based tool files
- project meetings to record progress or set targets
- project reporting
- day-to-day work assignment
- all project team "people" issues including staff appraisals and reviews

- day-to-day problem solving and troubleshooting
- liaison with any suppliers, both internal and external
- liaison with management
- liaison with the customer
- liaison with related/dependent projects
- normal ongoing recruitment

- quality
- quality plan/configuration management plan.

Though work by any member of the team is allowed for here, in reality a large proportion of this work (75–80 percent or even more) will be done by the project manager and/or team leaders.

Using this rule of thumb it is possible to work out the total amount of project management effort required in, say, man-

days or man-months.

By averaging this figure out over the life of a project it is possible to see what daily and weekly levels of project management are required measured now in hours per day or days per week.

The three examples below should illustrate this.



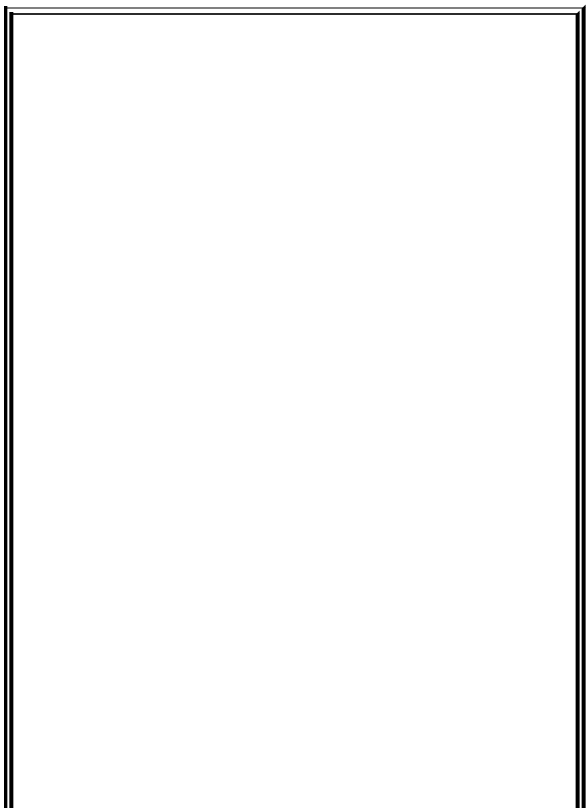
EXAMPLE 1

A project lasting 6 months and with a total effort of 22 man-months.

(Average number of people on the project is 3.7.)

Assuming 6 percent project management overhead based on total effort, this gives 1.32 man-months = 26.4 man-days (assuming 20 man-days = 1 man-month).

Thus, over the 120 elapsed days (assuming 20 working days in a working month) of the project, the project management overhead = $26.4/120 = 0.22$ days (= 1.76 hours for an 8-hour day) per day. So, in an 8-hour day, the project manager can reckon on this many hours per day being taken up with project management work.



EXAMPLE 2

A project lasting 12 months and with a total effort of 170 man-months. (Average number of people on the project is 14.2.)

Assuming 8 percent project management overhead based on total effort, this gives 13.6 man-months.

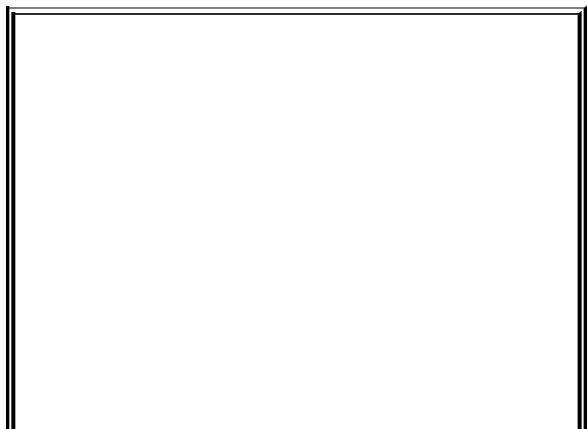
Thus, over the 12 elapsed months of the project, the project management overhead = $13.6/12 = 1.1$ months per month. So, in this scenario, a project manager is likely to be busy pretty much full-time doing nothing but project management.

If you don't allow yourself this much time and effort then you are not doing the job properly, and you are actually in breach of Step 3: while the

project may have a nominal project manager, in reality, nobody is doing the job.

The previous examples assume a flat team structure. In that case, a project manager carries the majority of the project management responsibility. An alternative to this is to set up some kind of team structure. This will have the effect of spreading the project management load across the project manager and the team leaders.

The example which follows takes [Example 2](#) and does exactly this, breaking the single monolithic project team down into three groups, where each group takes responsibility for one-third of the overall project effort.



EXAMPLE 3

For each team the total project management overhead is 8 percent of the total effort of 56.7 man-months = 4.5 man-months, which over the life of the project is 0.38 months per month, that is about 3 hours per day.

For the project manager who now has a team of three people, the overhead is 3 people for 1 year = 36 man-months by 8 percent, which gives approximately 3 man-months (2.88) or a quarter of his time over the year.

Note

This calculation assumes that all three team leaders are 100 percent as

effective as the project manager and that there is no overhead involved in maintaining this hierarchy. In reality the project manager would increase his project management overhead in dealing with the team leaders.

Examples 2 and 3 both illustrate valid ways in which this project could be managed. In both cases, you should always try to keep team sizes as small as possible. Then, in choosing between flat and hierarchical structures, there

are advantages and disadvantages of each. (There is no better or worse here ultimately it is the project manager's choice.)

MONOLITHIC TEAM OR FLAT STRUCTURE

The project manager gets to be totally hands-on. However, this requires a large amount of project management by the project manager, *which has to be allowed for in schedules.*

HIERARCHY OR TEAM STRUCTURE

Frees up project manager significantly to manage other projects and/or do technical work. However, the following points apply:

- 1. The project management effort required of the team leaders *has to be allowed for in schedules.***
- Handing over responsibility may initially involve the project

manager in more project management until she is happy that the team leaders can be fully relied on.

MAXIMIZE STRENGTHS

Or to put it slightly more negatively, but more bluntly: Minimize the risk of some turkey screwing up your project!

If you're lucky, you may be in a position where you can identify exactly the person you need to undertake each piece. When I say person, I mean that combination of personality, skills, experience, motivation, personal goals, strengths and

weaknesses that go to make up each of us. More often you will be given a group of people and be required to carry out the project with them.

I remember when I was a kid at school, we would have games sessions and teams would be picked. There was a handful of athletic superstars (or so they seemed to me) in the class, and from these would invariably be chosen the captains of the two sides.

Then the captains would pick individuals from a group of

solid citizens who made up the bulk of the class. Finally there would be five or six people left and the teacher would split these in two with his hand and allocate half to each side. I was always in this final collection! I mention this because the group you start a project with almost always are a mixed bag of individuals with diverse personalities, abilities, ambitions and motivations, and this is the group that you will be trying to mold into your team.

ASSIGNING PEOPLE TO JOBS

OK. You have your list of jobs from Step 2 of structured project management, you have got a group of people, some of whom are a spectacular match for particular jobs, some less so. You now need to assign the jobs to the people so that the project will get done.

What's the best way to do this? For any job on your list, and for any person in your team, the

following possibilities apply; the person:

1. can do the job and wants to do it

- can do the job and is prepared to do it
- can do the job and isn't prepared to do it
- can be trained/instructed into doing the job
- cannot do the job

Category 1

The first category is the ideal one. If you were to take only one idea away from this book, it is this: if, as part of your project, you can get a person doing what he or she wants to do, you have harnessed the greatest power on earth.

In this category, the person can do the job and likes to do it.

Later on, in [Chapter 17](#), we will talk about building a team, and I will say that the key question when interviewing somebody is

"What do you want to do?" If you can find people who want to do jobs that exist on your project you're guaranteed a Category 1 situation.

Category 2

The second category is still OK. You're happy that the person can do this job perhaps they've done it before, or you know it's within their capabilities. There may be more or less persuading to be done to convince them to do it,

and in [Chapter 15](#), I present an arsenal of weapons for you to use in resolving issues like this.

However, assuming that you arrive at a Category 2 situation, and can maintain that for the duration of the project, you should have no real problems. Let me also say though, as a note of caution, that there is a limit to how often you can persuade people to do things they don't really want to do. If what people are being asked to do doesn't fit in with their own personal plans then, in the long term, I

believe that you are swimming against the tide.

Category 3

For the third category, you've got a problem: the person can do the job, but won't. Maybe he's done it too many times before and is bored, maybe he feels he's not being paid enough. Who knows what the reason is? Basically what you've got to do here is to cause this situation to revert to either Category 2 (he's

prepared to do it) or Category 5 (for whatever reason, he's not going to do it, cannot do it). Again, you can use the techniques in [Chapter 15](#) to help you to do this.

Category 4

For the fourth category (can be trained/instructed into doing the job), then provided:

- you're prepared to put in the training time and money

- and you allow for training time in the project schedule
- and you allow for the extra management overhead
- and you're prepared to run with the risk that it might not work out

this category can often work out very well. In fact, you can often find yourself in a Category 1 situation because you may well have pushed the person into a more challenging job than she wanted to move

into.

Category 5

For the fifth category, you have a major problem. It may take you time to arrive at the conclusion that the person cannot do the job, but if you eventually do so, you need to find jobs within your project that the person can do. Failing that, the person belongs outside the project. In this balancing act of matching people to jobs you can

obviously get into problems of over- or under-utilizing people, and this is where something like a computerized project planning system is useful, though not essential. It may take several iterations before you get everything nicely balanced, that is:

- utilization of everyone less than or equal to 100 percent
- all training/instruction needs identified (note that this generates new jobs for

the job list)

- a feeling that, by and large, everybody will be happy with the jobs they've been assigned to do



CASE STUDY 5

It is interesting to read ([Huntford, 1993](#)) the different ways in which Scott and Amundsen assigned people to their jobs.

First of all there is a large difference in the numbers of people the two leaders took with them on their expeditions. Scott landed 33 people in total in Antarctica; Amundsen landed only ten. To some extent, this difference can be explained by the scientific side of Scott's expedition, but there is still a feeling about Scott's expedition of a general offensive whose aim was to win by strength of numbers. Amundsen's, on the other hand, smacks of a raid.

Scott took some people because they paid to join and his expedition

needed the funds. There is a certain amount of logic in his choice of scientists, but he still chose a large number of people with no previous Polar experience.

Amundsen, by comparison, chose his people like a craftsman selects a tool. He needed to get through the pack ice that surrounds the Antarctic continent in the summer-time; so, he took an ice-pilot, Andreas Beck. He planned to ski to the Pole so, apart from the fact that every man with him could ski moderately well, he brought with him Olav Bjaaland, who had skied for Norway at international level. Since he was using dogs he needed a skilled dog-driver. He had one in Helmer Hanssen.

The lesson for our own projects is clear: put together a small group of highly trained and focused people (as

opposed to a large group of generalists with a diffuse objective) and your chances of success are greatly increased.

ASSIGNING PEOPLE TO JOBS WITH FORM 2

The form shown in [Figure 4.1](#) may be of use to analyze the way you have allocated jobs to people. All 1s in the *Category* column would be ideal but a bit unlikely. A mixture of 1s, 2s and 4s is more likely, with any 3s and 5s being ones for particular attention. This form appears again in [Chapter 6](#) where the rightmost two columns are used.

Figure 4.1. Form 2, assigning jobs

Person	Job	Category	Trust	
			Yes	No

APPLICATION TO SOFTWARE ENGINEERING



[Figure 4.2](#) shows an example of the application of Step 4 taken from a real-life project. What the project manager has done is to overlay our form on a Gantt chart of the project. This is a neat idea because, for starters, it enables her to check that every job has been

assigned to somebody!

Figure 4.2.

Rostering pilot – Jobs	Person	Category	Trust	
			Yes	No
Project startup				
Contract finalization Sign contract	Tony Kenney	1	✓	
Pre implementation Implementation day	Cathy Kearny, Tony Kenney, Artwerk and a rep from each pilot	1	✓	
Software and hardware install				
Define hardware required	Cathy Kearny & Artwerk	1	✓	
Install hardware	Operations	2	✓	
Set up support modem	Operations	2	✓	
Install software	Cathy Kearny	4	✓	
Interfaces				
Personnel interface				
Design personnel interface	Cathy Kearny	1	✓	
Program personnel interface				
Ansos programming USA	Artwerk	2	✓	
Beaumont programming	Cathy Kearny	1	✓	
Testing				
Ansos testing UK	Artwerk	2	✓	
Beaumont testing	Cathy Kearny	1	✓	
Programming Part 2				
Ansos programming UK	Artwerk	2	✓	
Beaumont programming	Cathy Kearny	1	✓	
Payroll interface				
Design payroll interface	Cathy Kearny	1	✓	
Confirm design	Cathy Kearny and (Fin)	(2)	✓	
Program payroll				
Ansos programming USA	Artwerk	2	✓	
Beaumont programming	Cathy Kearny	2	✓	
Test payroll				
Ansos testing UK	Artwerk	2	✓	
Beaumont testing	Cathy Kearny and (Fin)	1	✓	
Programming part 2				
Ansos programming UK	Artwerk	2	✓	
Beaumont programming	Cathy Kearny	2	✓	

Rostering pilot - Jobs	Person	Category	Trust	
			Yes	No
Set up personnel interface				
Test Anso	Cathy Keary and (Per)	1 (2)	✓	
Test Anso and Beaumont	Cathy Keary and (Per)	1 (2)	✓	
Report problems	Cathy Keary	1	✓	
Program Anso UK	Artwork	2	✓	
Program Beaumont	Cathy Keary	1	✓	
Test and accept interface	Cathy Keary and (Per)	1 (2)	✓	
Implement personnel interface	Cathy Keary	1	✓	
Set up system in ward				
Data collection	Word and Cathy Keary	4		✓
Technical training UK	Cathy Keary	1	✓	
Controller visit	Cathy Keary, Word and Artwork	1	✓	
Data entry	Word	4		✓
Policies and procedure development	Cathy Keary and (Word)	1 (4)		✓
Reports definition	Cathy Keary and (Word)	1 (4)		✓
Reports generation	Cathy Keary and (Word)	4 (4)		✓
Scheduler visit	Cathy Keary, Artwork and Word	1 1 1	✓	
Test rosters	Cathy Keary and (Word)	1 (2)		✓
Staffer visit	Cathy Keary, Artwork and Word	1 1 1	✓	
Gerber alley interface				
Design interface	Cathy Keary	1	✓	
Confirm design	Cathy Keary and Paul Murphy	2, 2	✓	
Program MIS interface				
Anso programming USA	Artwork	2	✓	
Beaumont programming	Cathy Keary and Paul Murphy		✓	
Test payroll				
Anso testing UK	Artwork	2	✓	
Beaumont testing	Cathy Keary and Paul Murphy	1	✓	
Programming part 2				
Anso programming UK	Artwork	2	✓	
Beaumont Programming	Cathy Keary and Paul Murphy	1	✓	
Set up MIS interface				
Test Anso	Cathy Keary and Paul Murphy	1	✓	
Test Anso and Beaumont	Cathy Keary and Paul Murphy	1	✓	
Report problems	Cathy Keary	1	✓	
Program Anso	Artwork	1	✓	
Program Beaumont	Cathy Keary and Paul Murphy	1	✓	
Test and accept interface	Cathy Keary and Paul Murphy	1	✓	
Implement payroll interface	Cathy Keary	1	✓	

Rostering pilot – Jcas	Person	Category	Trust	
			Yes	No
Set up payroll interface				
Test Ansos	Cathy Kearny and (Fins)	1, (4)	✓	(✓)
Test Ansos and Beaumont	Cathy Kearny and (Fins)	1, (4)	✓	(✓)
Report problems	Cathy Kearny	1	✓	
Program Ansos	Artwerk	2	✓	
Program Beaumont	Cathy Kearny	2	✓	
Test and accept interface	Cathy Kearny and (Fins)	1, (2)	✓	
Implement payroll interface	Cathy Kearny	1	✓	
Set up catering rosters				
Implementation review	Cathy Kearny, Cote and Tommy Kenny	1 1 1	✓	
Data collection	Cote	4		
Controller visit	Cathy Kearny, Cote and Artwerk	1 1 1	✓	
Data entry	Cote			
Policies and procedures development	Kathy Kearny and Cote	1, 4		✓
Reports definition	Cote	4		✓
Reports generation	Kathy Kearny and Cote	1, 4	✓	✓
Schedular visit	Kathy Kearny and Cote and Artwerk	1, 1, 1	✓	
Test rosters	Cathy Kearny and Cote	1, 4	✓	✓
Staffer visit	Kathy Kearny, Artwerk and Cote	1, 1, 1	✓	
Catering pilot live				
Pilot Ward live				
Set up NCHD rosters				
Implementation review	Cathy Kearny, Tony K, and NCHD	1, 1, 1	✓	
Data collection	Cathy Kearny & NCHD	1, 4		✓
Controller visit	Cathy Kearny, Artwerk NCHD	1, 1, 1	✓	
Data entry	NCHD			
Policies and procedures development	Cathy Kearny and NCHD	4		✓
Reports definition	" " " "	1, (4)		✓
Reports generation	" " " "	1 (4)		✓
Schedular visit	" " " " and Artwerk	1 (4)	✓	
Test rosters	" " " "	1, 1, 1	✓	
Staffer visit	" " " " and Artwerk	1, 1		✓
NCHD pilot live	NCHD	1	✓	

Looking through the form it looks like we're in reasonable shape. There are some training requirements arising from the 4s and we should check that something has been allowed for these on the job list. If not we should add something. Also the 2s and 4s would want to be watched a bit more closely than the 1s. However, with luck, a lot of the 4s may turn into 1s with the appropriate training and supervision.

You can fill out one of these forms at a number of possible levels:

- for the project team as a whole
- for each person, as we have done above
- or for each person-job, as we talked about at the beginning of this chapter

The deeper you go the more insight you get. Also, you can do it at different times in the

project, say at the beginning of each phase. A reasonable compromise between not doing it at all and doing it repeatedly for each person-job might be to do it for each team member, either once (when they join the project) or, for very long projects, at the beginning of each phase.

Finally, one other thing that's useful is illustrated in [Figures 4.3](#) and [4.4](#): charts showing who is allocated to what project over what period. This helps you to keep track of multiple people across multiple projects,

and can also assist you with hiring plans. We have given two examples: [Figure 4.3](#) is for a single, large project; [Figure 4.4](#) is for a software engineering department.

Figure 4.3. Example of large project staff allocation

PROJECT	1998												1999												2000												2001			
	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3	4					
Feasibility Study	2	2	2	2	2																																			
Functional Spec.						8	8	8	8	8	8																													
Design								3	3	3	11	11	11	11	11	11	11	11	11	11																				
Implementation																																								
Integration																																								
TOTALS	2	2	2	2	2	8	8	8	8	11	11	11	11	12	12	15	19	19	33	33	33	33	33	33	33	33	33	12	12	12	12	12	12	12	12					

**Figure 4.4. Example
of software
department staff
allocation**

ACME SOFTWARE: STAFF ALLOCATIONS

GROUP/PROJECT	May 16	May	Jun	Q3	Q4	
BASE TECHNOLOGY (Fred)						
Management		1	1	1	1	Fred
Project A		3	3	2	2	Roy, Steve
Project B		11	11	12	12	Jill, Greg #1, Greg #2 etc.
Working in H.Q.		1	1	1	1	Eleanor
		16	16	16	16	
PRODUCTS (Steve)						
Product A		2	2	2	2	Alan, Kathy
Product B maintenance		1	1	1	1	Noel
		3	3	3	3	
CUSTOMER SERVICES (Milton)						
Management		1	1	1	1	Milt
High-end products		5	5	5	5	Tracy, Lisa, Sid, Mike, Bert
Low-end products		1	1	1	1	Jim
Mid-range						New hire
		7	7	7	7	
ADMIN. & MGMT. (Alex/Jim)						
General Manager		1	1	1	1	Bert
Personal/Admin		5	5	5	5	George, Tammy, Niki, Viki, Lee
		6	6	6	6	
CUSTOM SERVICES						
Technical Director		1	1	1	1	Mark
Software Dev. Mgr.		1	1	1	1	Alan
Pre-sales		1	1	1	1	New hire
EC Products		6	8	8	8	Lou, Joe, Al, Robin, Tony, Dick
US Products		5				Gary, Harry, Anne, Tim, Davin
Major customer projects		2	3	3		Steve, Linda
Available			4	5	8	
		16	18	19	19	
TOTALS		48	50	51	51	

PSI CONTRIBUTION



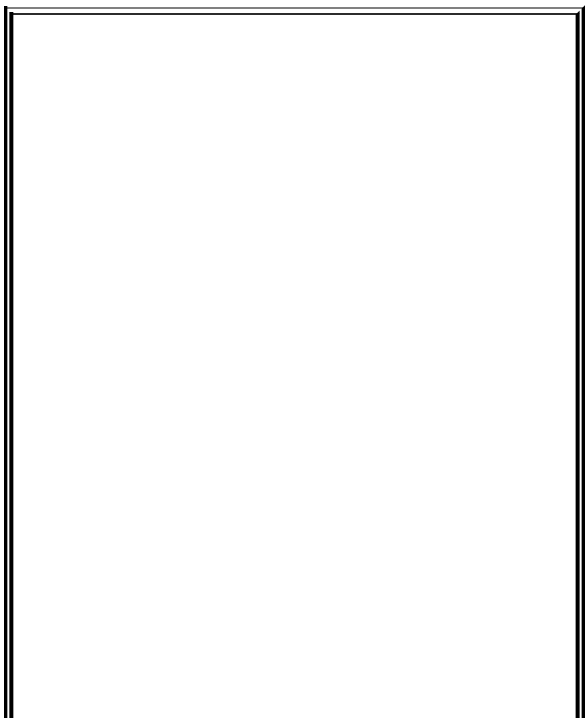
With a maximum PSI of 10 you get your points on Step 4 as follows:

- If every job on your list has a human being's name against it award yourself a PSI of 3.33. As we said, it may be that early on in a project you don't know who all the people are, but you should be striving to find

out as quickly as possible. A score lower than 3.33 here will (hopefully) keep you focused on the fact that there are issues here which haven't been resolved.

- If you take people's other commitments, including your own and especially your project management one, into account award yourself 3.33.
- If you do the maximizing the strengths analysis that

we described earlier award
yourself a PSI of 3.33.



STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- **MAKE A LIST OF THE JOBS
THAT NEED TO BE DONE**
- **THERE MUST BE ONE LEADER**
- **ASSIGN PEOPLE TO JOBS**

Chapter 5. STEP 5: MANAGE EXPECTATIONS, ALLOW A MARGIN FOR ERROR, HAVE A FALLBACK POSITION

INTRODUCTION

CONTINGENCY: ALLOW A
MARGIN FOR ERROR, HAVE
A FALLBACK POSITION

MANAGE EXPECTATIONS

A WORD ON COMMITTING

APPLICATION TO
SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

By the time you have completed Steps 1 - 4 you have actually finished the planning for your project. If you have carried out the steps as we have described, then what you have done is you have built a working model of how your project might unfold over time. In a sense you know the *answer*. You know what's going to be delivered, you know when it will complete, you know how many people you need and a

whole load of other useful information. In theory now, you could come scampering out of your office, go in to your boss or customer, and announce "We know the answer. It'll be November 27th," or whatever.

You could. However, stay your hand for a moment while you consider one other thing. What will you do if it doesn't turn out like your model predicts?

You see it's all too easy especially if you've gone into things in excruciating detail like we talked about, and even

more so if your plan is now being output by a PC-based project planning tool to forget that everything that's in the plan is a *prediction*, a guess, a shot in the dark, a gamble. You know nothing in your plan for a fact. In fact the only thing you do know for a fact is that *your plan is definitely not right*. It couldn't be, it's a prediction of the future. Given that this *is* the case, the most you can hope for is that you get a lot of it right, and that where it's wrong it's not too far wrong. You need

to ask yourself the question "what will I do if it doesn't work out like this?" And the answer you have to give yourself is to build in some contingency. That's what the first part of Step 5 call it Step 5a is about. It's about allowing yourself a margin for error, a fallback position.

Having built in some contingency for yourself, then you need to sell your plan to the powers that be and create the expectation to which you will then deliver. This is the other part of Step 5 Step 5b.

We describe each of these steps in turn.

CONTINGENCY: ALLOW A MARGIN FOR ERROR, HAVE A FALLBACK POSITION

Explicit or hidden contingency

If you were a project manager and you worked in construction, then on your plans people would expect to see a line item called *Contingency*. It might be set,

for example, at 15 percent of the project budget and/or elapsed time. If people didn't see this, they would assume one of two things: the more charitable of the two would be that it was an omission on your part; the other would be that you had taken leave of your senses. They would believe this because in omitting contingency, it would imply that you felt confident you could predict the future.

There are some software companies regrettably few, I have to say where a similar

ethos applies. The management and the customers of these companies expect to see contingency built into plans. Not doing so is regarded as project management practice somewhere in the range of improper to criminally negligent.

So what happens in the other software companies? Well, in these, contingency is actually what the management or the customer takes the red pencil to first. "OK, now where can we shorten this baby [or make it cheaper]?" ponders the Big

Cheese, the Great Gorgonzola.
"Ah, contingency, you won't
need that for starters."

If your organization is one of
the enlightened ones, then put
in the contingency explicitly,
and plenty of it as much as
you can get away with.

If, on the other hand, your
organization is of the red-pencil
variety, then you need to hide
the contingency. PC-based tools
are very powerful in helping
you to do this quickly. Another
useful way of doing it and
they'll never spot it is to put

in fake activities. You can couch these in technobabble (there are numerous software packages that will generate buzzwords and acronyms for you) and so you have activities like:

- Update table lookup
- Memory maintenance
- Changes to existing
- Decision on tool
- Re-work

You know the kind of thing I mean just find some applicable to your particular application area and you can reuse them time after time. Of course it's better if you can put your cards on the table and set out the contingency explicitly, but failing that you have to get it in there somehow.

Not so long ago, I did some consultancy for a company. It essentially ended up with me spending an afternoon working with a project manager helping him to hide contingency in an MS Project plan. It was his

management who were paying my fee and here I was preparing material to, in a sense, hoodwink them. I have to tell you that I felt entirely morally justified in doing what I was doing, on the basis that one day they would thank me for it.

They did too. About a month ago; when their project came in on time and within budget the time and budget that *included* the contingency.

Finding contingency

You can use the four parameters we first identified in [Chapter 1](#) to help you build in your contingency. These are:

- Functionality
- Delivery date
- Effort (cost)
- Quality

Functionality

Some projects get structured such that on some happy day in the future, somebody will pull the great lever and the whole system will power up and work perfectly. (The nuclear deterrent system operated by the superpowers during the Cold War was such a system! I had a friend who ran a small software house and he was a veteran of more payroll system installations than you could shake a stick at. Based on his experience of these, he was confident that there could never be a nuclear war: he'd

never seen a payroll system work right first time, so why should thermonuclear Armageddon be any different?) Either it all works perfectly on the promised delivery date or ... it doesn't work at all. That's how some projects are conceived.

You can certainly organize your project this way, but then you're really asking for trouble. You've given yourself no room for maneuver, no fallback position.

As an alternative to this you

can do what is called phased delivery. Is there some minimal set of functionality your project can deliver which will get your customer off the ground (start them doing useful work) while at the same time familiarizing themselves with, and debugging the system? Then, can you add some more functionality and some more and some more? At any point, if you find you've reached for a milestone too far, you can always fall back on the last version you delivered.

Delivery date and effort (cost)

You can add contingency using either of these merely by adding on an additional percentage to allow for contingency. Typical numbers tend to be in the range 10–15 percent, but I suspect the only significance of these numbers is that they tend to be what people can get away with. As a general rule, I would say the more you can get the better. Certainly the more chance you

have of your model being seen to be right and hence, of having a successful project.

You can add the contingency:

- on a blanket basis, across the entire project
- on a per phase basis, that is, to each phase
- to those jobs on the critical path

Also, it's probably better to aim for adding a percentage onto

the delivery date, because in doing so, it often means you can hold onto people for that extra period of time, and so achieve an even bigger contingency on the effort.

Quality

If you can measure the quality of your project deliverables, then you can use these as bargaining counters in the contingency game.

For example, in the

development of a software system you might use mean time to defect (MTTD) as a measure of quality. (For an excellent discussion on this, see [Puttnam and Myers \(1992\)](#).) Then you might aim for an MTTD of, say, 2 days, but in a pinch you'd be prepared to settle for 1 day or 1.5 days. Again, you gain some room for maneuver, some margin for error.

MANAGE EXPECTATIONS

OK, you've built the model of your project using Steps 1 - 4. You've built in contingency using Step 5a. Now you're ready to go to the powers that be, be it your boss or your customer, and tell him the answer. Typically, this is the point at which you announce the answer and your boss or customer does the equivalent of slapping their thigh and saying "you've got to be joking. It's going to be *when?*" Or "You

want *how many* people?" And so on. I'm sure you've been there before; and, if not, if you've just joined this business, then it'll only be a short space of time before you will be.

So now what do you do? Well, for starters, let's be quite clear that the one thing you don't do is to agree to whatever *he* proposes. That is unless your model shows it is possible.

The way forward is conceptually simple. It may well, however, require plenty of

guts on your part. Here's what you do.

The model shows one possible way that your project could work out. If you either (a) anticipate that the powers that be won't like that way, or (b) find out they don't like it when you tell them, then you can use the model to generate other possible scenarios. How do you this? Easy, by varying our four trusty parameters:

- Functionality

- Delivery date
- Effort (cost)
- Quality

So you might, for example, ask what the effect is of adding more people. Or see what (reduced functionality) can be achieved by the delivery date that Sales have already promised. Or see if you can get people to buy into an extension on the delivery date. Or you can try combinations of these. At all times you use the model

you have developed to generate new possible scenarios think of them as flavors to the basic vanilla model. Using the model keeps you honest. It stops you from signing up for something that is patently impossible.

If you don't use the model if you merely succumb to pressure then you are doing precisely that. You are signing up for a mission impossible. You are opting for a quiet life now and trying to put out of your mind the certainty that a bomb is going to fall on you when the

chickens come home to roost as they inevitably will, because your model says that they will.

The one thing you should not do in this game is to sign up for a mission impossible no matter how much and how severe the pressure that is applied to you. And, your model will act as your protection. Often, in the traditional way that this game is played, the negotiation becomes an emotive confrontation between two sides. Using the method we propose, it becomes a logical

discussion of facts. The Roman writer Sallust said, *ubi res adsunt, quid opus est verbis*, "when the facts are at hand, what is the need of words?"

People can be as emotive as they like in our version of the game and it doesn't matter. The models you develop will coldly and logically show what can and cannot be done. Hold your ground and ultimately the good guys should win.

Another stratagem that the powers that be may adopt will be to question your model, to try to chip away at the

estimates contained in them. "It couldn't possibly take two weeks to do that," they will say, "you're overestimating everything here." Your answer to this is reasonableness itself. "They're only estimates," you will say, "so they're definitely not 100 percent right. They may indeed be too high." Pause for dramatic effect. "But they might also be too *low*. We don't actually know. We think there's a fair chance, because of the detail we've built in (via Step 2) that they're accurate, but they are only predictions. Why

don't we run it for a couple of weeks and see how things turn out? Then, when we make our final decision, we've a better chance that it's the right one."

The key to all of this is discussion, negotiation, trying to keep the whole thing on a civilized level. Later in this chapter (see the application to software engineering section), we offer some additional tactics you can adopt to try to keep things on this civilized plane.

Ultimately, however, there may just be someone out there

crazy enough to say "I don't care about models, figures or anything else. Just do it, or else." If that happens, you should take your courage in your hands and walk away from the whole thing. A glib statement, you may say, with mortgages to pay and families to support, but what's the alternative? The alternative is that somewhere down the line you're sitting in a blackened smoking ruin and you're being spotlighted as *the* person responsible for it all.

Come on, life is too short for

that kind of unhappiness. And anyway, there are still enough unsuccessful projects out there that good project managers are still in demand. So let's just summarize the process:

- Steps 1 - 4 build the basic vanilla model
- Step 5a adds contingency to the vanilla model
- Step 5b creates a number of possible *flavors* of the model. It is one of these and only one of these

that your boss or customer can buy

- Once the boss or customer has chosen the one that he wants, this becomes the one that you try to make happen. This is the expectation you create
- Finally, the chosen option is documented so that it is clear to everybody involved who signed up for what.

A WORD ON COMMITTING

Sooner or later in projects we have to *commit*. In doing this, we basically make a promise and then try to make that promise happen. If everything turns out as we predicted, then we are heroes; otherwise our name is mud.

The first thing that should be said about committing is that you should try and leave it to as late as possible in the

project. The longer you can get away with not having to make this promise, the better for all concerned. Each day that passes you find out more about the nature of the project, and all of these details, when fed into a model of your project, increase the likelihood that the promise you make will come true.

Sooner or later, though, the fateful day will come and you will have to make the promise.

It is both my belief and my experience that if you use the

planning techniques described in Steps 1 - 5, then there is a fair chance that the project will just run through to a successful completion. If so, then fair play to you and God bless us all.

If this doesn't happen, however, if you find that you have eaten up all your contingency and that there is no way that your promise can now be met, then you've got to come clean. Our instinctive reaction is perhaps to do the opposite: to cover the whole thing up, pretend that nothing is wrong and hope it'll all be alright on the night. Not

to put too fine a point on it,
this is madness.

Coming clean won't be
pleasant. On my courses I liken
it to exploding a 500 lb bomb
around everybody. But again,
what's the alternative? The
alternative is that blackened
smoking ruin we spoke of
earlier. Think of it as
exchanging a 500 lb bomb now
for one of about 20 megatons
later!

So you come clean. You carry
out Steps 1 - 5 again. You make
a new commitment, set a new

expectation and off you go. Ultimately your conduct will be seen for what it was: professional project management.

Some final thoughts on this subject.

- Know what your customer is expecting. What he is expecting will be based entirely on what you have led him to believe. Can you trim back from this without affecting him unduly? Sure you can. Check and see

what he could actually put up with at a pinch. Be sure that your development schedule enables you to deliver this first.

- Build a number of demonstrations and/or prototypes into your schedule so that your customer can get her hands on the system early and you can verify that you have read her expectations correctly.
- See what is required to go

beyond her expectations and plan to go there if at all possible that'll really wow her. But it will only wow her if you have given her all of what her basic expectations were.

- Don't spring surprises on your customer. Customers are very forgiving people once they are treated as intelligent human beings. It is only when they get the impression that you are not being straight with them that they react badly.

- Some things have no margin for error. Know which things on your project are like this, watch them like a hawk, and make sure they happen.
- A retreat to a fallback position need not be an ignominious thing; it can be presented as a master stroke of planning and forethought another example of managing expectations.



CASE STUDY 6: MANAGE EXPECTATIONS

Scott left England in a blaze of publicity. The British Empire was at its height. To any rational outside observer it was clear that the British would be first to the South Pole. Amundsen slipped quietly away, and it wasn't until he was long beyond anyone's reach that his objective was made known to the world. If he failed, then he would disappear quietly into oblivion. If Scott failed, then it would be a very public failure.

CASE STUDY 7: ALLOW A MARGIN FOR ERROR

Scott moved one ton of supplies out onto the path he would follow to the Pole. These depots were indicated by cairns of snow and single marker flags. Scott allowed just the amount of food and fuel that he needed.

When at the last minute he switched the number of people in the Polar party from four to five, this put all his calculations awry. A hastily drawn up plan was put in place to offset this problem, but Scott and his companions were to die of starvation, scurvy (caused by vitamin deficiency) and exposure.

Amundsen put four tons of supplies on his path. His depots were marked by sets of marker flags radiating out from the depot. Each marker flag had

a note attached to it indicating the distance to, and direction of, the depot. On his return, Amundsen's men had put on weight and left two tons of food behind them in depots.

CASE STUDY 8: HAVE A FALLBACK POSITION

We referred previously to the opening of the Battle of the Somme during the First World War, when the British launched an offensive which they hoped would end the war ([Macdonald, 1983](#)). The British commander's plan was admirable. Quite simply, he hoped to give his men a walkover.

He would do this by submitting the Germans to the biggest artillery barrage of the war, or indeed of any war. It would last continuously for a week. At the end of that time, his men would quite literally walk across to what would remain of the German lines. Well, of course, there was no walkover. The German lines were very much intact and nearly 60,000

British soldiers were killed or wounded on that one day alone. Over 19,000 of them were killed, most of them in the first couple of hours of the battle.

The British project plan was based on the premise that there would be a walkover. When that failed to materialize, there was no contingency plan. You are doing one of the most risky things imaginable if you take a project forward on the basis of no contingency plan, no fallback position.

It's human nature: you tell somebody you're going to do something, you do it and they're pleased. You do more than what you said give them an added bonus and they're delighted. But fail to do what you said and your name is mud. Once you've made your plan

you need to look at it and ask:

- What are people (you, the project team, your customer, your boss, your peers, the rest of your organization, people outside the organization) expecting to happen when the project completes? Can you deliver on what they are expecting?
- What if something goes wrong? Spend a while analyzing what can go wrong. Use brainstorming or some of the techniques discussed in [Chapter 11](#), to see how well you are set up to deal with these eventualities. There are some things upon which you will have no margin for error, and these become crucial jobs for you.

Many things, however, are less critical and you can perhaps live without them for a while.

Identify the critical and less critical things. Do these cause people's expectations to be modified?

- Is there a fallback position? If something disastrous happens can you still save your skin and the project? If you can't then at least you go forward consciously knowing the risks you run; but often there are things you can do.

You should do this for the project as a whole and for each job in your job list. Try to

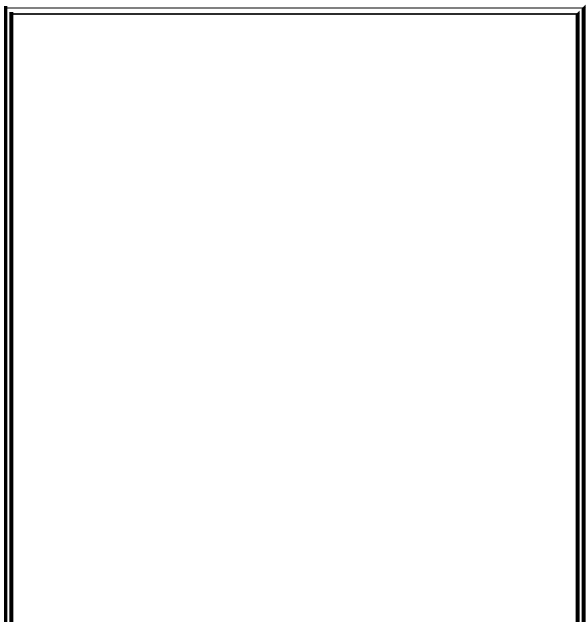
categorize the risk associated with each job as high, medium or low. Then focus on the high-risk ones and make sure, if nothing else, that these have a margin for error and a fallback position. We have said that managing expectations, allowing a margin for error and having a fallback position are all facets of the same issue. There's an argument that says:

- define your goal
- identify a fallback position

- set expectations equal to the fallback position
- then your margin for error is everything between the fallback position and the real goal

There is some merit in this approach. The big danger is that the fallback position now becomes the goal; so that you are still left with a situation where you have a goal and no fallback position. Here's a final example from history showing all three things manage

expectations, allow a margin
for error, have a fallback
position as facets of the
same issue.



CASE STUDY 9

The "unsinkable" (expectation) *Titanic* was launched in 1912. She could carry 2,000 – 2,500 passengers and crew and had lifeboat capacity for 1,100 (no margin for error). When she struck an iceberg on the night of April 14, 1912, over a thousand people lost their lives (no fallback position).

APPLICATION TO SOFTWARE ENGINEERING



We promised you some additional tactics to help protect yourself from predatory bosses and customers while at the same time keeping things civilized. Here they are.

Bosses or customers often want to know at the beginning of the

project when it is going to end and what it will cost. Often you are also required to bid fixed-price for a job when you are very early in the product development life cycle. What do you do in these cases?

I would maintain that the only time you can accurately estimate a project, that is commit to a fixed schedule and effort, is when all the design is finished. At that point I would be quite happy to give a fixed-price estimate to a customer (in other words, to pin my ass on a particular delivery date

and staffing level). *Before that you can't.* I repeat, before that you can't. If you are being pressurized to commit earlier, then there are a few stratagems you can employ.

Two fixed-price quotes I'll give you a fixed-price bid for the design phase and at the end of that I'll give you a fixed-price bid for the rest. You'll have your design so that if you don't like my bid for the implementation you can take the design elsewhere. Another way of looking at this is "I can only give you a date/effort by

which the design will be done. At that point I can give you a date/effort for the rest."

Parkinsonian estimating

Often pressure is applied by telling you the end date by which the system must be ready and the size of the team you're being given to deliver. A Parkinsonian estimate, worked back from the end date, can show the impossibility of what you are being asked to do. It can be done very quickly. You fit your six (or whatever number you use) phases back from the end date. Typically the

effects you get are things like: to make the deadline the development should have started two months ago; the deadline can only be achieved if coding takes two days; or, the deadline can be achieved if we don't do any testing.

Load the estimate If you are put in a position where you have to give a fixed-price quote, then you must load the estimate to give yourself sufficient room for maneuver. And it's OK to tell the customer you're doing this and why. "We don't know how big the job is,

Mr Customer, so given our estimates, see, and our level of knowledge at the moment, we have to assume worst case scenarios, so this is the reason for our pricing. If, at the end of the design phase, we find that we overestimated we'll be more than happy to pass the reduction on to you. Oh, and by the way if anyone is telling you differently ... they're wrong."

You have to stick to your guns. If you have done your estimates properly and believe in them, then you hold the moral high ground. If you

commit to something you cannot achieve then you will preside over a catastrophe and you will be left looking fairly silly.

I don't know of anyone who has been fired for refusing to budge from their estimates. I do know of numerous instances where people were fired for screwing something up!

Play the change control game If you are forced to make a fixed-price bid, be sure you deliver it with all the relevant fine print. State

explicitly, and in excruciating detail, what was and wasn't included and the assumptions you made. Then as other items come to light in the course of the analysis and design phases, you can play the change control game where every new item becomes a change to the contract.

Tell 'em it's impossible Here you take on the project but none of the associated guilt or responsibility. Nice prospect, huh? In this scenario, somebody asks you to take on what looks like an impossible

mission and you say yes! Not yes unconditionally, however, but a yes with certain provisos.

Imagine the scene. You've looked at what they're asking you to do, and your model and all associated reason tells you that it's impossible. They're adamant, however, it's been promised — it has to be done. So you make the following offer. You say: "I've looked at what you're asking me to do and I believe that it's impossible for reasons that I've outlined to you. However, given the squeeze you're in I'm

prepared to do whatever I can to help. Here's my offer. I'll take on this project and run it for you. I can't guarantee you I'll make those commitments, but I will guarantee you that as soon as I know *for a fact* that the commitments are impossible, I'll let you know. Then you can decide how you want to present that to your customer."

There now. They can't accuse you of not being a team player. Note too, the repeated use of the word *you* during this little speech, ensuring that the guilt

stays well and truly where it belongs.

PSI CONTRIBUTION



Contingency, or margin for error, can be thought of as the gap between your fallback position and your actual goal. Here the maximum PSI contribution is 10.

You get a score out of 5 points on Step 5 if you've built some contingency into your plan. The more the contingency the higher the score out of 5.

You get the other 5 points if you can show that the expectation you've created lies somewhere between the fallback position and the goal. The closer the expectation lies to your fallback position, the higher your score out of 5.

If your project has no margin for error, then you lose 15 off your cumulative score here.



STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- MAKE A LIST OF THE JOBS
THAT NEED TO BE DONE**
- THERE MUST BE ONE LEADER**
- ASSIGN PEOPLE TO JOBS**
- MANAGE EXPECTATIONS,
ALLOW A MARGIN FOR ERROR
HAVE A FALLBACK POSITION**

Part 2: REVIEWING AND IMPLEMENTING THE PLAN; ACHIEVING THE GOAL

INTRODUCTION

At this point in the proceedings you have spent all your time planning your project; you

have still not set out on the journey. It is not an exaggeration to say that the fate of your project has been, to a very great extent, sealed by the work you may or may not have done so far. If you have applied the steps of structured project management to your project, then you will:

- Know in intimate detail the goal of the project

- Have a plan very detailed for the immediate future, less so thereafter, but at least showing the remaining important milestones
- Have identified who is going to be the leader
- Have assigned the members of your team to the jobs on the list of jobs
- Have ensured that

everyone understands
the big picture and
how their particular
piece fits into the
jigsaw

- Know what everyone is expecting and have some room for maneuver between that and what you plan to deliver

You can now begin the journey secure in the knowledge that you are starting from a secure

base, and that you have done as much as you could do at this stage to expect the unexpected. If you have not done some or all of these things, then there is a startling amount of very bleak news for you, which I had better begin to recite. If you have tears to shed, prepare to shed them now. Your project may not fail. After all, life is full of surprises that's part of what makes it so enjoyable.

If your project does not fail, then you will have pulled off a major success. You will have succeeded against the odds. I'll bet it won't have been very pleasant and that there were times when it was just awful, and I'd say nobody enjoyed themselves all that much. But if you did it, hats off to you is what I say. I do wonder why you would choose to put yourself and your team through an experience like that. And I

also wonder why you choose to do projects in this way, since it can't get any easier from one to the next, never knowing whether it's going to work out or not. Anyway, it's your call and good luck with the next one.

That was the good news. The bad news is that I believe your project is bound for disaster, speeding toward it as surely as the *Titanic* toward its encounter with the iceberg. Your project

will be a disaster and you, as project leader, will have a monumental mess on your hands that will make you wish you had never become involved with the project. Whatever you had hoped to gain from it (money, promotion, prestige, an enhanced résumé, fame, recognition, whatever it was) will be lost in the appalling shambles that lies ahead of you. There are only two possible pieces of advice I can offer

you. Either do a proper plan or get out while you are still untarnished.

Assuming you have done a proper plan, with all of the detail that we've stressed so much, then what you have done is you have identified one possible way that the project could unfold. What we are now going to do, in Steps 6 - 10, is to make this plan into a self-fulfilling prophecy. We are going to try to *force reality to adhere to the*

plan. And if that sounds as though you would have to have god-like qualities to make it happen, forget it, you need nothing of the sort.

The plan says that certain sets of jobs are going to have to get *bitten off* every month, every week and ultimately every day. If the jobs are bitten off exactly as specified in our plan in other words, if we reach the monthly, weekly and daily targets that our plan specifies

then the project will remain on schedule.

Amongst other things we can use the interaction of our four parameters:

- functionality
- delivery date
- effort (cost)
- quality

to try to ensure that this happens. Thus, for instance, if a particular

milestone has to be met on a certain day, then we can, say, add people (increase effort) or reduce functionality to try to ensure that this happens. If we do this for every milestone, large and small, throughout the life of the project, then the project will remain on schedule, and our plan will indeed turn into a self-fulfilling prophecy. Reality will indeed have adhered to the plan. These comments apply

completely too, to cost or budget.

This is what Steps 6 – 10 as well as the material in [Part 3](#), about running multiple projects, is all about.

There is another thing on offer here as well. At one point I wanted to call this book "[The Lazy Project Manager](#)," but Viki, my editor, wasn't at all keen on the idea. The word *lazy* is normally used in a pejorative sense, but in

this context I intend it to be highly complimentary. The Lazy Project Manager doesn't just have successful projects. She has them *by doing the least amount of work possible*.

You see, the Ten Steps don't just guarantee a successful project. They also are *the least you have to do for a successful project*. Mathematicians would describe them as a "necessary and sufficient condition" for a successful

project. *Necessary* because you have to do these things; *sufficient* because these things are enough, once you have done them you need do no more.

The Ten Steps stop you from fretting about whether you have done enough project management. They stop you from waking in the middle of the night in a cold sweat wondering whether you have missed something. They tell you

when you have done enough, when you have done your job as well as it can be done.

In [Chapters 610](#) and in [Part 3](#), I will show you how you become a Lazy Project Manager.

Finally, there is one last use for the Ten Steps worth mentioning. When I first began managing projects, and when I took over or started a new project, the first thing I did was to "read myself

in." This involved laying my hands on every document that I could find related to the project, everything related to the technology the project used, and any other background information that seemed remotely relevant. Then I would read and highlight away to my heart's content. Often the stuff was non-existent or out of date, and then you ended up with an incomplete or skewed picture of what the project

was all about.

With the Ten Steps,
reading-in is at an end.
The first five steps tell you
exactly what things you
need to know:

Where can I find a
description of the goal of
the project and evidence
that the detail in this goal
is being developed (specs,
designs, etc.)?

Is there a plan? (If there
is you can use the
material in [Part 4](#) to

analyze it.) Who's leading the project if it's not meant to be you?

Are all jobs assigned to people? Have people's other commitments been taken into account?

Where are the potential weaknesses in the work assignments?

Is there contingency and a fallback position? What expectations have been created? How do the goal, the fallback position and the expectations all relate

to one another?

APPLICATION TO SOFTWARE ENGINEERING



At this stage you should have the following:

- a detailed description of the goal of the project (from Step 1)
- a job list, complete to

the first milestone,
and as detailed as you
can be thereafter, but
with the major
milestones included
(from Step 2)

- a project leader (from Step 3)
- an allocation of your team to the jobs on the job list, so that all jobs are covered (from Step 4)
- some contingency

plans (from Step 5) if things go wrong

In other words, you have all the things that a good project plan should have, and you are ready to set forth into the great unknown. Just as a final cross-reference, here's a proposed table of contents for a project plan, into which you could structure all the information you have gleaned by applying Steps 1 through 5.

SECTION IN PROJECT PLAN	INFO. FROM STEP
0 Related documents	
1 Introduction Project description	1
Statement of work	1/2
2.1.1 Requirements specification	
2.1.2 High-level design	
2.1.3 Low-level design	

	design	
	2.1.4	
	Implementation	
	2.1.5 Alpha	
2	test	
	2.1.6 Beta test	
	Deliverables	1
	2.2.1 Software	
	2.2.2	
2.2	Documentation	
	2.2.3 Services	
	2.2.4	
	Acceptance	
	criteria	
	Completion	
	criteria (how do	
2.3	we know when	1

it's over?)

Development plan

Work
breakdown
structure
(WBS)

2

3.1.1

Requirements
specification

3.1.2 High-

3.1 level design

3.1.3 Low-level
design

3.1.4

Implementation

3.1.5 Alpha test	
3.1.6 Beta test	
3.2 Work plan	2
3.3 Effort	2
3.4 Schedule	2
3.5 Milestones	2
Resource	
3,6 loading/project ramp-up	2,3,4
3.7 Budget	2,3,4
3.8 Critical items	5
3.9 Project organization	3,4

Resources required

required

4	4.1 People	3,4
	4.2 Equipment	2,5
	4.3 Training	2,3,4,5
	4.4 Other	2 5
5	Assumptions	1 5
6	Unresolved issues	1 5
	Appendix A	
	Derivation of estimates	2,4,5
	A.1 Effort and manpower	
	A.2 Schedule	
	A.3 Margin for error	5

Finally, software projects fail for a variety of reasons, but often they fail just because:

- things took longer than expected
- requirements kept changing in an uncontrolled way
- something came up that nobody had anticipated
- nobody estimated

in any kind of a
commonsense way
how long the thing
was likely to take or
what it would cost

- technical problems,
that nobody had
foreseen, caused
delays

Admittedly, some of these
things can be valid. But
often they are presented
to cover up an absence of
basic planning and
forethought. Carry out

Steps 1 5 of structured project management on your project and you give yourself some kind of a fighting chance. If you inherit a project, insist that you be allowed to carry out these steps, and don't be surprised if your management are alarmed at the result.

However, don't you be alarmed. Structured project management provides you with the proof that what you say is true, and anyway, at this

point in a project, you have the moral high ground. Hold your nerve and if the project is important enough, they'll let you do it the correct way. This is also true if you are dealing with a subcontractor. Ensure that, at the very minimum, his proposal contains the elements already outlined, and kick him out if it doesn't.

You have done all your planning and your project has hit the road. The next

five chapters talk about what you need to do during the journey to your destination.

STEPS 15

SIGNIFICANCE OF PSI AT THIS POINT



Two points need to be made here. First, you should notice that 70 percent of the PSI is accounted for by Steps

1 5, the planning steps. If this is true, and all the evidence we have gathered to date suggests that it is, then this confirms what was said earlier about the fate of your project being sealed in the planning phase.

Secondly, the projects we have analyzed to date reveal that 40 is the threshold from the planning steps. What I mean by this is the following. When the project starts, during its

very first moments of life, things like the goal (Step 1), the list of jobs (Step 2) and so on will register little if anything in terms of scores. For example, you might have little more than a one- or two-line sentence describing the goal of the project, and this would earn you a score of perhaps 1 or 2 out of 20, if you were lucky. Similarly for the other four steps.

The 40 threshold says that the first things you have

to do in the project, *the absolutely first things you have to do to the exclusion of all others*, are to go through the processes of:

- Defining the goal in successive levels of detail. (If you think about it, the first few phases of any project life cycle requirements gathering, design, and so on are concerned with doing precisely

this.) (Step 1)

- Refining the plan as the goal becomes more definite and detailed. (Step 2)
- Ensuring that one person is driving the project, setting up the project organization structure and installing the processes and procedures that are going to operate throughout the

remainder of the project. (Step 3)

- Getting your hands on people (real warm bodies) and allocating them to jobs. (Step 4)
- Structuring your plans such that you have a fallback position and a margin for error. (Step 5a)
- Creating the right expectations. (Step 5b)

In doing these you will gradually raise your PSI scores until they reach the critical 40 level. Until you get to 40, you should be trying to expend as little effort as possible on the project, since you won't be sure of anything like a successful outcome until you get past 40. Also, doing anything else, other than the things mentioned above, is wasting time, effort, resources, money and energy.

Chapter 6. STEP 6: USE AN APPROPRIATE LEADERSHIP STYLE

[INTRODUCTION](#)

[THE LAZY PROJECT
MANAGER](#)

[APPLICATION TO
SOFTWARE ENGINEERING](#)

[PSI CONTRIBUTION](#)

INTRODUCTION

Your team is moving forward; a mixed bag of individuals as we discussed previously. Some seem to be megastars, some are good solid players, some look decidedly dodgy. There may only be a small number of them, so you are like a close family; or there may be thousands of them. Each person is an individual with his or her own wishes, fears, desires, prejudices, skills, experience, relationships,

problems, ambitions. How can you even begin to set about leading such a complex organism? How can you allocate work? Find out what's going on? Keep a check on morale? How can you even measure morale?

"Beats me," is probably the most accurate answer to these questions. *It is an incredibly difficult thing to do.* It is probably safe to assume that at any given time some of the people in your team will be unhappy. It is probably equally safe to assume that at any

given time some of the people will be behaving in such a way as to contribute nothing or even to retard your project.

The best any of us can do here is to apply those management talents we have as best we can. There are no rules here, only guidelines; and I present mine under the general heading of *use an appropriate leadership style*.

Each job on your list will be carried out by a person. This makes the combination of job and person unique, i.e. the

same job would be done differently by different people. This means there will be hundreds or perhaps thousands of unique job person events happening on your project, all of which you have to manage. The sum of how you handle all these events is what makes up your leadership style.

Each of us will have a basic style autocratic, democratic, soft, hard, etc. but we will have to vary our style depending on the job person event that we are dealing with. Does this mean we have to

have hundreds of different styles? And if not, how do we know what styles there are and when to use them? Here's how. In [Chapter 4](#), we discussed the five situations that could arise with regard to a person and a job. These were:

1. can do the job and wants to do it

- can do the job and is prepared to do it
- can do the job and isn't prepared to do it

- can be trained/instructed to do the job
- cannot do the job

There is another concept we need to introduce at this point. This is the idea of *trusting* somebody to do a job. Let us use the following definition. If, when you ask somebody to do a job, you can regard the job as being done, then we will say that you trust that person. Obviously if this applied to all the people and all the jobs on your project, there would

hardly be a need for you at all, the whole thing would go swimmingly and complete according to plan. It is precisely because you generally do not trust some or all of the people on your project that you are needed, and that projects go awry.

What would cause you to trust somebody as we have defined it above? You might trust them because you had worked with them before and knew they could be relied upon to carry out any job. Or perhaps other people whom you trust might

have spoken highly of them. Or maybe you would try them out on your project: tell them what you expect from team members that when you give a person a job you want to be able to assume it will be done and try them out with two or three tasks. If they perform then you can start to build up your trust in them. A good test is to ask yourself whether you feel confident entrusting your reputation to their judgment and performance. Another is that you can identify some solid evidence as to why you believe

you can trust them.

If they cannot pass these tests, then you don't trust them.

Simple as that. As the project progresses, you may come to trust them. That's a different issue. Also you may trust them on some jobs but not on others. I repeat the test if you don't feel you can entrust your neck completely to them or have no solid evidence, then for the purposes of this discussion, you don't trust them.

[Table 6.1](#) shows the ten possible scenarios and the

leadership style I suggest you adopt in each case. Where it says "Resolve into a (2) or a (5)," we will discuss methods in [Chapter 15](#) whereby you might do this.

Table 6.1.

CASE	TRUST	DON'T TRUST
1. Can do it/likes to do it	(A)	(B)
2. Can do it/prepared to do it	(A)	(B)
3. Can do	Situation	

it/won't do it	same as (E)	Resolve into a (2) or a (5)
----------------	-------------	-----------------------------

4. Can do it with training	(C)	(D)
----------------------------	-----	-----

5. Cannot do it	(E)	Resolve into a (2) or a (5)
-----------------	-----	-----------------------------

(A) You trust the person to do the job. Perhaps they've done it before or if not something similar to it. They like doing it or at worst they're putting up with doing it. In short, they're the experts and can make all

the decisions internal to the job. Leave them to it. Don't concern yourself with it, other than to tick it off your list on the day when it completes on schedule. Even if something does go wrong we will catch it because of *No Surprises* as discussed in the next chapter.

(B) They're doing it, their hearts are probably in the right place, but you're not super-confident that they'll do it right or on schedule. You have to watch them: your neck depends on it. There's no need to bug them to distraction a

bit of gentle hand-holding will do the trick. If there are decisions to be made then you need to work with them to arrive at the correct conclusion.

(C) They've done other things well in the past and now you're trying them on something new. They've come through on everything so far, but this is a new ball-game. Another case of watching them gently, as in (B). Remember that your neck depends on it. You may have a decent relationship with them anyway from which the trust has arisen and know what

level of monitoring to apply. Decisions can be made democratically as in (B).

(D) They've never done it before and you have no reason to believe they can or will do it. It's back to school, guys. Hand-holding. Inching forward with mini-goals spelled out in detail. Constant monitoring for trouble signs.

(E) Here we have a problem. In fact we have two problems. First, there is the problem of what will happen to the job that this person was meant to

be doing. This job isn't happening: we need to make alternative arrangements about it. Then there is the problem of the person. On at least one of the jobs on our project this person isn't performing. This potentially means there are others where this is also, or will become, the case. We need to determine what we are going to do about this person. But as project managers, especially a Lazy Project Manager, our first priority is the job. First we have to get that squared away. Then, as a secondary priority,

we can worry about the person.

You can use Form 2 from [Chapter 4](#) which is reproduced here ([Figure 6.1](#)) to understand which management style to use in which job-person situation.

Figure 6.1. Form 2, assigning jobs

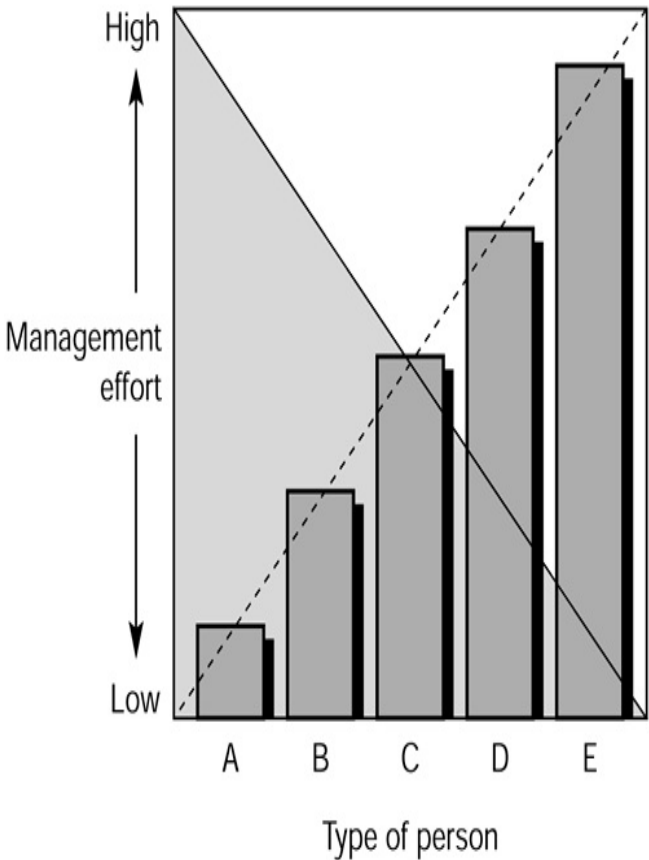
Person	Job	Category	Trust	
			Yes	No

THE LAZY PROJECT MANAGER

Let's for a moment rate these five different styles on the following scale as shown in [Figure 6.2](#). The vertical axis is a measure of how much management time and effort you put into a particular task. It is a measure of how much micromanagement you do, how much you are in your people's faces, how much you stick your nose into their business. We have rated each of our five

styles (A) (E) against this scale.

Figure 6.2. Different management styles



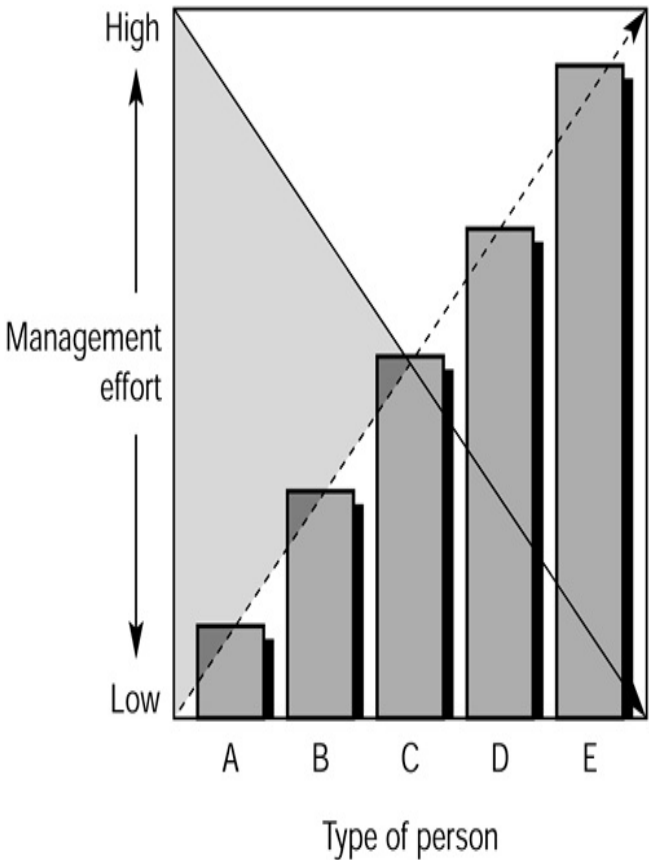
The *ascending dotted* line shows the gradual increase in management effort as we move from leadership style (A) through style (E). Now it seems to me that as warm, living, loving human beings, our natural tendency would be to allocate our time and effort in a way that is represented by the *solid descending* line.

We would enjoy spending time and effort over on the (A) (B) end of the spectrum as shown in [Figure 6.3](#). Over here is the happy uplands where all good

men agree. Over on this side are competent, motivated people. Milestones are being achieved and commitments being met; it's good news, high morale, forward progress; all that great stuff that we dream will happen on our project. We could spend happy days over here and go home in the evening thinking that the world is indeed a fine place.

Figure 6.3. Where the project manager should spend least

time



Over on the (D) and especially (E) end of the spectrum, on the other hand, is the twilight zone of project management. Here is the land of permanent bad news, cock-ups, screw-ups, SNAFUs; having to tell bad news to bosses and customers; having to eat humble pie; having to give people bad performance appraisals or no salary rises; having to fire people; having to coax people painstakingly through things; having to confront, annoy, argue, and generally dissipate vast amounts of emotional

energy trying to keep things on the rails. Over here is feeling that you have been let down by people. We want to be here, in short, like we want a hole in the head!

And so (I don't know if you've been guilty of this, but I certainly have) we spend our time on the (A) (B) end and procrastinate, delay and generally avoid doing things on the (D) (E) end. We spend time having really interesting conversations with the good guys and in the process wasting our time and theirs;

and we avoid dealing with issues which are rotting away inside our project.

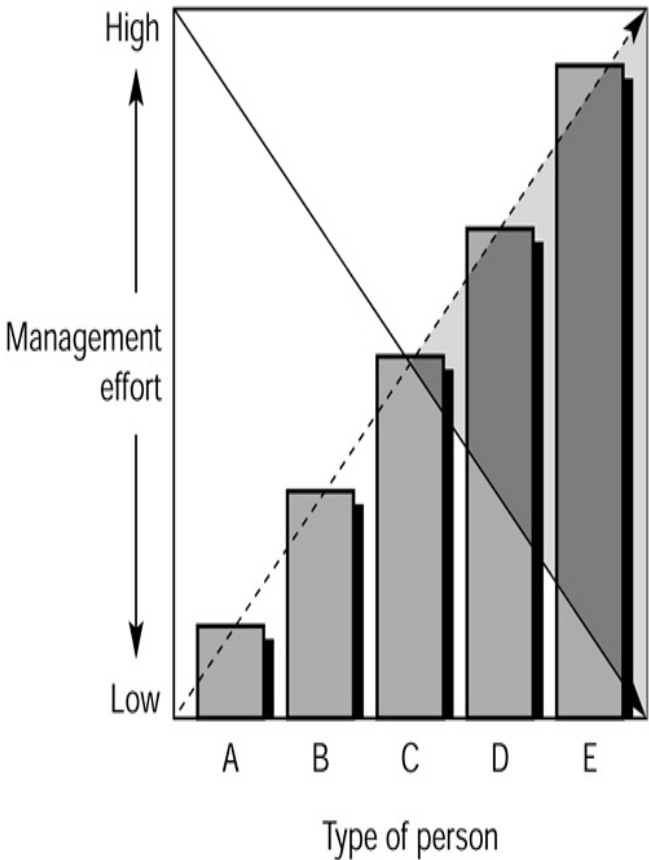
The Lazy Project Manager has no time for such foolishness or dithering. She knows that spending time in the shaded zone, while it may be enjoyable, *is of no value whatsoever in progressing the project.*

Indeed, if you spend lots of time trying to *manage* people who should really be treated with an (A) (B) style, not only is it a waste of time and effort,

it can actually be counterproductive. Such people end up feeling that you don't trust them, their expertise and their judgment, and they can become very demotivated with lots of what they perceive as interference.

Instead, the Lazy Project Manager puts in the effort where it counts: in the shaded zone shown in [Figure 6.4](#). Here every ounce of management time and effort contributes directly to the forward progress of the project.

Figure 6.4. Where the project manager should spend most time



The Lazy Project Manager becomes an object of complete bafflement and irritation to her colleagues because she seems to have an uncanny knack of knowing how to leave well enough alone, and to get in where the problems are. In short, the Lazy Project Manager not only always has successful projects but, in the process, she makes it all look like a piece of cake.

APPLICATION TO SOFTWARE ENGINEERING



Here we fill in Form 2. We have filled out the form categorizing a team of people in accordance with the scheme introduced in [Chapter 4](#). From the form it looks like we're in reasonable shape. There are some training requirements arising from the 4s; however, it is the rightmost

two columns which interest us. This form shows exactly where the project manager ought to expend management effort and suggests the management style that will be right in any particular situation.

In [Figure 6.5](#) we see that everyone appears in the *don't trust* column of the form; the reason might be that this is a brand new team with which the project manager has not worked before, and everyone has been classified as either a Category B or a Category D. This team will require intensive

monitoring on the project manager's part. There is no part of the project he will not have to stick his nose into. If, after a period of time, he finds that he has come to rely on some people, and finds that they consistently deliver, he may have reason to revise his rating of them and place them in the *do trust* column.

However, until then he will have to keep a careful eye on things to make sure nobody drops the ball.

Figure 6.5. Using Form 2

Person	Job	Category	Trust	
			Yes	No
Snow White	Project manager	1		B
Grumpy	User interface	4		D
Sneezy	New functions	4		D
Dopey	Programmer	1ish		B
Sleepy	Programmer	1ish		B
Happy	Designer	1		B
Bashful	Database specialist	4		D
Doc	Testing	4		D

PSI CONTRIBUTION



If your style tends to be static, either constantly hands-on or always hands-off, give yourself a low score here. If you vary your style with the circumstances, well done, and give yourself a high mark. Maximum PSI contribution is 10.

--

STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- **MAKE A LIST OF THE JOBS THAT NEED TO BE DONE**
- **THERE MUST BE ONE LEADER**
- **ASSIGN PEOPLE TO JOBS**
- **MANAGE EXPECTATIONS, ALLOW A MARGIN FOR ERROR HAVE A FALLBACK POSITION**

Implementing the plan; achieving the goal

6. USE AN APPROPRIATE LEADERSHIP STYLE

Chapter 7. STEP 7: KNOW WHAT'S GOING ON

INTRODUCTION

USING YOUR PLAN AS
INSTRUMENTATION/THE
LAZY PROJECT MANAGER'S
DAY

POSITIVE SIGNS

APPLICATION TO
SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

In this chapter we talk about three things:

- 1. Using your plan as instrumentation to guide the project. In doing this we will describe how the Lazy Project Manager spends her day;**
- Positive and negative signs on a project;
 - The doctrine of *no surprises*.

USING YOUR PLAN AS INSTRUMENTATION/THE LAZY PROJECT MANAGER'S DAY

We're all familiar with instrumentation. In a car, for example, we have a series of gauges, clocks and other display devices that give us quantitative information about what is happening out in the world. We then take actions based on this information.

We can use our plan in exactly the same way to give us information about what is happening on our project. We can then make sensible decisions and take appropriate actions to control the progress of our project toward its goal. Fancier project management books than this one call this process project monitoring and control.

It is in this process that all of the hard work that went into Steps 1–5, and notably Step 2, begins to pay off. Note too the way the plan we developed

during the planning steps has multiple uses:

- The plan we developed with Steps 1 through 5a enabled us to estimate the project;
- Applying Step 5b to this plan enabled us to generate versions of it from which our management and/or customers could pick;
- Now we are going to use the chosen version again as the instrumentation with which we will guide the

project.

The myriad uses we make of our plan are yet another reason why it is well worth putting time and effort into producing a good one. And we see yet again the central position of planning in having a successful outcome to our project.

You can use your plan as instrumentation whether it is on paper or stored on a PC-based planning tool. Everything I am going to say here will

apply in either case. The advantage of using a PC-based tool is that you will get results quicker and your work will not contain the errors that will almost certainly creep in if you do things manually on paper. I am going to illustrate things here by reference to a PC tool, but I reiterate that everything I say here can also be done manually.

Let us assume that you have your plan and you have entered it into a PC tool such as MS Project. Every morning when the Lazy Project Manager

comes into work, she fires up her PC and brings up the plan on the screen. Now, the plan may well have many activities in it. Four to five hundred wouldn't be uncommon for the levels of detail we have said you should produce.

But remember the format of our plan. It contains all the major milestones and then is very detailed to the next milestone. So when we look on our screen, and we look at those jobs clustered around today, we probably won't see more than a few dozen that fit

within, say, a week either side of today. Then, when we look to the right on our screen, we will see those other jobs which are part of the big milestones off in the future.

The Lazy Project Manager then has two things she has to do, before she, quite literally, has finished her project management for the day. These two things are (1) check on the current stuff, and (2) look off into the future and try to unearth more detail about those future milestones. We discuss each of these in turn.

Check on current stuff

The few dozen jobs clustered around today give the Lazy Project Manager a to-do list for the day.

First there are the finishers. These are the jobs that the plan says should have finished. Have they? Go and see. Is there a deliverable you can hold in your hand? If so, then they're done. If they're not done, then you update your plan to reflect this fact. This change causes a ripple through

the rest of the project. We will discuss this ripple in more detail in a moment.

Next there are the starters. These tasks should have started. Again, have they? Is somebody beavering away on it, doing exactly what the plan says? If so, fine. If not, what *are* they doing? Does the plan need to be changed? Or do they need to be refocused?

Finally there are the jobs that are in progress. Depending on the leadership style you are adopting in particular situations

(see [Chapter 6](#)), you may want to go and check on some of these. On the other hand, you may be happy to leave the people to it. Note that this is the Lazy Project Manager at work again.

All of these items may generate additional off-line activities (meetings, memos, e-mails, faxes, phone calls, and so on) that you have to do. These are your project management tasks for the day.

Open up the surprise

packages

Each day that passes you find out more information about the project. Almost like a detective gathering clues, you get more and more information about things which lie in the future, and which up to now had been hazy, or guesses, or assumptions, or just plain unknown.

Having checked on the current stuff, you now look through the stuff that lies in the future and try to fill out any additional

details that you can. Remember that what you are trying to do is to achieve the *man day* level of detail in all of your plan, so that you can say what each person on your project is doing for each day over a particular period of time. Once you have added all the detail you can, then you're finished.

Doing the two things just described may have a number of possible effects on the schedule. These are:

- no change

- a slip which can be recovered
- a slip which can't be recovered

Let's look at these in turn. The first is trivial. It means that you have managed to keep your project on schedule for another day. You can ask no more. Go home you've done your project management for today.

In the second case, the changes in the schedule

generate a slip. However, by using our plan as instrumentation, we can do things like move people around, work some overtime, drop some feature which was a bit of a frill or similar actions to bring the project back on course. Note that those four parameters of the Apocalypse:

- functionality
- delivery date
- effort (cost)

- quality

show us what our options are. Note too how our model, which we have put such a lot of work into, is serving us well again. Our model tells us what is and what isn't possible. It stops us making foolish assumptions such as the one that says we can always recover from any slip just by working harder — a well-worn recourse of software practitioners. As Caesar said, "Men usually willingly believe what they wish."

The third possibility is that we have a slip and we can't recover from it, no matter how much we tweak our four parameters. Now, we have a problem. This means that we have run out of contingency. The only thing that can help us now is a lucky break, and we might indeed let things go for a few days, a week, a couple of weeks to see if such a thing happens. Remember that our model is still only a prediction and so, occasionally, the gods smile on us and things take less time and effort than we

anticipated. Occasionally! It may be we get such a break which puts things to rights.

However, if at this stage things are not improving this technique I'm describing is the way some people, including myself, manage their overdraft! you have to come clean with the powers that be.

Will this be pleasant? Almost certainly not. Humble pie rarely is. *But you have to.* The alternative is unthinkable. You can be caught in a small explosion now or a

thermonuclear one further down the line.

Go on. Go break the news to your boss and/or customer. Whichever of these situations occurs, once you have worked your way through the business we have just outlined, your project management, your project monitoring and control, your know-what's-going-on, is finished for the day. You can do no more. You have done everything that needs to be done. You can go home and rest secure in the knowledge that it is all under control. Even

if things went wildly awry, you caught it at the earliest possible moment. Now, that's a professional project manager for you!

POSITIVE SIGNS

The following are all indicators that should be regarded as positive signs on a project.

People are having fun

[Roland Huntford's account \(1993\)](#) of Amundsen's return journey makes exhilarating reading. Amundsen knows he has won, all he needs to do now is return his team to safety and announce his

achievement to the world. He has food and fuel aplenty and for the last 200 miles marker flags that he deployed during his outward journey now make the return trip more like an extended ski race. Doing 20-30 miles per day (three to four times what Scott is doing) his men career into Framheim, their base, on January 26th, 1912. It is a good sign if your project gets to a point where:

- people are strutting around a bit

- their confidence is up
- morale is high, pretty much independent of working conditions on the project
- they feel the worst is behind them
- their belief in their own abilities and those of the team, has increased
- there is less tension and ill-temper

- there are more practical jokes
- people enjoy being in the team environment; in short, people are having fun
- milestones are being met

Amundsen laid out his plan in terms of daily milestones that all his team could understand: each day they would travel 20 miles, a quarter of a degree of latitude, and stop. Thus every four days they were a degree

closer to the Pole. Also as soon as they had done their 20 miles, they could stop.

Milestones completing and being ticked off is a good sign of progress as well as being a significant morale booster, since it implies that the estimates that were made before the project got under way appear to have been good ones.

Independent feedback

You may receive an early indication from a source

independent of the project team that the project is working. *Independent* is the key word here. This is sometimes called "putting your fingers in the wounds." In the case of a project to develop a product, this could mean a demonstration version, a prototype, an early release, or a partially working system. There was a period in Operation Desert Storm where the general public in the West were receiving little or no such independent feedback from the Allies, with the result that we

were very unclear as to how that particular project was progressing. (*Note:* (before you write to complain!) I first drafted this particular chapter of the book in mid-February, 1991. At that time the land war in the Gulf had not yet started and it was unclear exactly what was going on.)

People leave the team alone

This one is more from the project team's point of view. If

the project is going well, there will appear to be fewer outside interruptions, people sticking their noses in, meetings, and so on.

Life after the project

Again, more from the team's point of view: people start to talk of a life after the project ends. They have been bound up with it for so long but now they can begin to see a light at the end of the tunnel.

Common vision

The goal has stabilized; there is complete accord and no confusion as to what the ultimate objective is.

Few crises

Nothing appears to have been forgotten; there is very little firefighting; everything is calm and efficient.

Negative signs

The following are indicators, any one of which should make a project leader extremely nervous about a project.

Milestones are not being met

Things aren't working out according to plan. Estimates appear to have been whacky. Milestones which were reported as done somehow become undone. The same milestone gets rescheduled several times.

"Everything's under control"

It's a negative sign if somebody, particularly the project leader, uses this phrase!

Low morale

High staff turnover or sickness. People looking dour or moaning a lot. An apparent absence of anything that could be termed *team spirit*. Nobody appears to be enjoying things very much.

A loss of confidence in the team.

Personality clashes

These would be worrying particularly if they occur in the upper echelons of the project's management.

Nobody's having fun

A really extreme instance of this is Scott's return from the Pole which makes appalling

reading: it takes colossal effort from his weakened and demoralized men to move what have now become light loads, a few miles. Nerves are racked as meager supplies run low and depots are difficult to find. One of the team, P.O. Evans, dies. Oates, the horse handler, is suffering from an old leg wound he received during the Boer War. He too dies; he walks out of the tent, never to be seen again. Finally, 14 miles short of a supply depot, Scott and his two remaining companions, held up by a blizzard, pitch

camp. They are never to set forth again.

Procrastination

You ask about particular jobs several times, and each time you find an excuse why the job hasn't been done. An interesting historical example of this is General George B. McClellan's command of the Army of the Potomac during the early part of the American Civil War, where Lincoln kept asking him to advance on the

Confederate Army and McClellan kept finding reasons not to (this earned McClellan the sobriquet *The Virginia Creeper!*). See Cotton (1960, 1961, 1963).

Goalposts keep shifting

The goal has never stabilized. Each time you thought you had identified it, it got changed or expanded in some way. Note that this can happen through the process of *creeping elegance*: lots of small changes

to the goal are made. Each one in itself is small and can probably be accommodated without too much difficulty. However, the sum of all the changes represents a dramatic change to the goal.

Mistakes are being made

Lots of things seem to have been forgotten; mistakes become more frequent; there's a lot of carelessness about.

Lots of crises

There appears to be a crisis every other day. Panic meetings are commonplace. Firefighting is the order of the day.

No independent feedback

All you have is the project leader's assurances that things are under control. There is nothing you can see and feel for yourself.

The slough of despond

A feeling, either among the team or the customer, that there is no end in sight. A good historical example of this was the Battle of Passchendaele fought between June and November 1917 in Flanders ([Macdonald, 1978](#); [Terraine, 1977](#)). In popular memory it has become the classic First World War battle. The summer of 1917 was one of the worst on record, and the rain turned the low-lying Flanders land into a sea of mud in which men

literally drowned.

The Battle of Passchendaele actually consisted of a number of smaller battles and the result, from the infantryman's point of view, was that time after time he was asked to cross a churned-up morass in the face of withering machine gun and artillery fire to attack German lines protected by vast belts of barbed wire. Those men who survived one battle were soon thrown into the next one, with the result that men lost hope. The war would go on for ever. They would all die, it

was just a question of when and better to do it sooner rather than later and avoid the awful suffering of living and fighting in a quagmire. If your project gets to the point where people are repeatedly being asked to work overtime, put in one more effort, come in at weekends, and if they feel that no real progress is being made, then your project is in trouble.

The rumor machine

The normal grapevine that

exists in any organization suddenly gets replaced by a rumor machine, and that machine seems to be spewing out wilder and wilder rumors as the days go by.

Things are not as expected

As you have planned your project you will have gone through the process of visualizing every step of the journey. As a result of doing this you will be expecting

certain things to be a certain way, that is there will be signs you will be looking out for and expecting to see. If you do not see them or if the signs are not what you expected, this could be a cause for concern. An example of this from history is the Battle of the Somme. Part of the aim of the artillery bombardment (which was discussed in [Chapter 5](#)) was to cut the enemy's barbed wire entanglements. It was confidently predicted that these would be cut by firing shrapnel. However, on the eve of the

infantry assault, scouting parties were reporting large tracts of barbed wire intact. These reports were ignored by headquarters staff being put down to nerves with the resulting disaster that we have already described.

"No surprises"

Run your projects on the basis of *no surprises*. This means that there should only be one real crime: that is, to be aware of some impending problem on

a project and not to alert anybody to the fact. This applies to anyone connected with the project: leader, team or customer. Mistakes can and will be made and this is all allowed for, but to keep some time-bomb hidden, for whatever reason, is to cause a major threat to the project and to let down the other people involved in it.

APPLICATION TO SOFTWARE ENGINEERING



How do you know what's going on? On a large software project there may be scores or even hundreds of people beavering away on components of the system. These components may run into many thousands, all of which have to fit together in an intricate and perfect way. How

do you know how well the system is coming together; how complete is it and how much remains to be spent to complete?

As well as all the qualitative things outlined above, there are a couple of very quantitative ways you can track progress on your project. The most obvious one is, assuming that you have your plan on a computer-based project planning tool, that you can now track progress by marking off jobs done, adding in new jobs and changing existing ones.

The advantage of the tool is that it is unforgiving. If there is a slip it shows you, in a depressingly clinical way. Correspondingly though, it illuminates the critical path the way of recovering from slips if recovery is possible.

The whole plan can be laid out for you in as much or as little detail as is relevant to your particular interest and the analysis you want to do. As I've said earlier, anyone who runs any kind of a project, no matter how small, without the use of one of these tools is nuts. An

even more sophisticated way of tracking progress, particularly on large projects, is through the use of the earned value concept. You will find a good discussion of this in *Software Engineering Economics* ([Boehm, 1981](#)). In addition, the particular project planning tool you are using will explain how it has implemented the concept, and what earned-value-based reports and analyses are available.

Basically, earned value allows you to determine whether the costs incurred on a project up

to the present time are in proportion to its current level of completion. In other words, it tells you if you are spending more than you planned to spend given the work that has been completed up to this time. It also enables you to predict what the project (or any job within it) will cost at completion if the current trend continues. Remember, however, that earned value is not a panacea. Get many of the other things right and earned value is useful, effective and gives you a very fine finger on the pulse

of your project. But if the basic, simple stuff that we have been describing in the rest of this book isn't carried out, all the earned values or fancy financial reporting systems in the world won't save you.

PSI CONTRIBUTION



Step 7 has a PSI contribution of 10. If you do what we have said and keep your finger on the pulse of your project then you get a high score here. Otherwise, if from day to day, you're out of touch with what's happening, score low.



STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- MAKE A LIST OF THE JOBS THAT NEED TO BE DONE**
- THERE MUST BE ONE LEADER**
- ASSIGN PEOPLE TO JOBS**
- MANAGE EXPECTATIONS, ALLOW A MARGIN FOR ERROR, HAVE A FALLBACK POSITION**

Implementing the plan; achieving the goal

6. USE AN APPROPRIATE LEADERSHIP STYLE

- **KNOW WHAT'S GOING ON**

Chapter 8. STEP 8: TELL PEOPLE WHAT'S GOING ON

INTRODUCTION

STATUS REPORTS

THE LAZY PROJECT
MANAGER'S WEEK

A VARIATION ON THE LAZY
PROJECT MANAGER'S
WEEK

APPLICATION TO

SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

In this chapter we talk about status reports, and I tell you how the Lazy Project Manager spends her week.

We are told that knowledge dispels fear. Amundsen and Scott provide conflicting examples of how to handle information dissemination within a project. Amundsen explained the objective and his plan to achieve it. The plan was explained, discussed and posted

up in the mess for everyone to see. As we have seen, the plan enabled people to visualize progress and set a finite limit on what they would have to do each day.

Scott by comparison would remain unclear as to his plan until well into his journey to the Pole. Whatever plan he eventually evolved was not shared with anybody until the moment when people had to know. At the last moment he changed one of the cornerstones of the plan: the number of people who were to

go to the Pole with him. Even his backup plan, such as it was, was hastily put together, and the result was that when he got into serious difficulty, whatever hope he had of rescue was ruined by the poorly laid and confused plans he had given to his subordinates.

What I am saying here is that people are thinking creatures. Everyone has inputs to give, everyone has elements of creativity and everyone feels more self-worth if they know what's going on. People will operate at their best if they

know what the big picture is and how their component, no matter how trivial, fits into it. One assumes this is particularly true in a military situation if you are asking people to make extraordinary and sometimes final sacrifices for you.

So Step 8 is *Tell people what's going on*. Having said this I will now proceed to contradict it. There is the concept of the leader acting as a filter. Think of it as a fort with the project team inside and the leader acting as a sentry on the gate. The world outside is in a

constant state of change. During the life of the project, some of these changes will have implications for the project and the team. The leader needs to determine how much of this change to let through to the team.

Some change has to be let through. Some is healthy and, as I have already said, the team will feel more self-esteem for knowing that they are being treated as intelligent people. However, there will be changes that might, or definitely would, cause a major problem in the

progress of the project if the team got wind of them. These changes you must either:

- block
- or failing that, censor so that their threat is reduced
- or finally failing that, relay to the team in such a way as to minimize the effect on them

and you have to do all of this without hiding true realities from them. A tall order? You

bet! The foregoing is equally true of outgoing information. There will be ups and downs on a project, and the outside world needs to be apprised of progress. But there may be things which occur on the project that you must:

- stop from going outside
- or failing that, water down so that when they get outside, their true effect is reduced
- or failing that, transmit so

as to minimize the effect on the outside world

and all of this without hiding true realities. All of this is *a very tall order*. What this book does is to try to give you an armful of methods and ideas that you can apply in just such circumstances. In [Chapter 15](#), there are a whole bunch of ideas that you can use. In any particular situation you will be able to apply more than one of them with differing results. It will be up to you to pick the one that suits you and the

situation best.

The analogy is with a carpenter's toolbox. Pretty much anything in it — a hammer, the handle of a screwdriver, the flat side of a plane — will drive a nail into a piece of wood, but some of the tools are better suited than others. So too with the tools in this book. Several will achieve the effect. It will be up to you to judge which effect is best for you.

STATUS REPORTS

It's funny the things that seem to be constant across lots of different cultures, and one of them, I've discovered, is that during their school days, many people have written an essay entitled *A Day at the Seaside*.

Perhaps you did too, and if so, maybe you remember how it went. All human life was there the weather, the personalities, the events, the emotional highs and lows, the

interplay of characters and, at the end of it all, we tumbled into bed, sand between our toes, badly sunburned, covered in calamine lotion, tired but happy after a tremendous day.

I mention this because most status reports seem to bear striking similarities to a day at the seaside essays. They might as well be called "Here were all the dead interesting things that happened on the project last week." As we read on excitedly we find out all the minutiae that occurred on the project and invariably there is the

happy ending just like in our essay; the upbeat conclusion that says everything's OK, all's right with the world.

But is the project on target? Hey, I don't know. And you certainly won't find out from reading a day at the seaside status report. If you want to come away with a permanent warm feeling, then a day at the seaside is just for you. But if you want some useful information, you will have to look elsewhere.

The status report I propose is a

top-down creation, going from the most high level to the most detailed. It says in no uncertain terms what the status of the project is; and you can perhaps see why people might not want to write these — they are completely unforgiving and clinical in terms of what they reveal.

Here, then, is a structure you might follow in writing a status report.

Level I

Is the project on schedule? Yes or no? Your status report template could have two boxes where you tick one. If the project is on schedule, people may not want to know any more.

Level II

At this level you present some high-level information on the state of our four parameters: functionality, delivery date, effort and quality.

Functionality

A couple of sentences stating at a high level:

- what the functionality started off at
- any major changes that have occurred

and pointing at the change control log (see [Chapter 1](#)) where the detailed changes and history of changes can be seen.

Delivery date

Here's what it was originally	mm/dd/yy
Here's what it is now	mm/dd/yy
Here's how it changed	nnnn

Change no.	Date	Reason chang
1	mm/dd/yy	xxxxxxxxxxxxx>
2	mm/dd/yy	xxxxxxxxxxxxx>
...	...	...
n	mm/dd/yy	xxxxxxxxxxxxx>

Effort

Here's what it was originally	nnnn man-days
-------------------------------	---------------

Here's what it is now	nnnn man-days
-----------------------	---------------

Here's how it changed	nnnn
-----------------------	------

Change no.	Date	Reason chang
---------------	------	-----------------

1	mm/dd/yy	xxxxxxxxxxxxx>
---	----------	----------------

2	mm/dd/yy	xxxxxxxxxxxx>
...	...	...
n	mm/dd/yy	xxxxxxxxxxxx>

Quality

If you are measuring quality, say, with something like mean time to defect (MTTD), then you could also treat it the way we treated delivery date or effort, above. If you're interested you'll find a good discussion of MTTD in [Puttnam and Myers \(1992\)](#).

Finally, at this second level, you should make some general statement about the state of health of the project. Even though it may be on schedule there may be some calamity looming. Here you can say in a few sentences what the big issues, known risks and problems outside your control are, and what you are doing/ would like done about them.

Level III

At Level III, if you're feeling a

bit like a frustrated novelist because you haven't been able to get into a day at the seaside, then this is your chance for a bit of narrative. If you like, you can write down the things that are currently happening. This would correspond very much to the "current stuff" we talked about in [Chapter 7](#). Do it by person maybe, and that way everyone gets something said about him and feels loved and wanted.

Level IV

Finally, at this level, you can throw in everything and the kitchen sink. Here you can include a copy of the current plan the Gantt chart showing all the detail in the world.

Finally, the last question is who do we give this great opus to? Well, in the spirit of what we said in the introduction to this chapter, it seems to me that we should give it to as many people as possible. Clearly there may be things that we don't want the team or the boss or the customer to see, but

there's nothing to stop us doing a sort of "vanilla" report for general consumption with a little note attached if there are things we want to report to a specific audience.

By giving it to the team, we enable every person to see the big picture and how their part fits in. Other people may spot things that we haven't seen. The more problems are identified now, the fewer problems we'll have down the line.

Equally, by giving the status

report to bosses and customers we avoid the project becoming a black hole where money is poured in, and someday something may come out. We not only do a good job but are *seen* to do a good job.

As an example of what I mean here (and this is very much by way of illustration rather than a rule) on a project I ran for a client, I produced two versions of the status report. One the same one went to the team and my boss, i.e. *my* client. The other went to my client's client, i.e. my customer. There were

two differences between the reports.

- 1. All references to who was working on what were removed from the customer's version on the basis that this was my client's business and not the customer's.**
- Any downbeat or ominous messages were removed from Levels II and III of the customer's version. This, however, was because the project was generally going

well. If it had been otherwise then we would have been passing those messages on to the customer. Had we had to do this, then, among other things, we would have been preparing the customer for bad news. And in doing this, we would have been following the principle of no surprises that we talked about in Step 7. If we had had to break bad news to the customer, what he would have seen all the way would have been professional project management of the situation.

THE LAZY PROJECT MANAGER'S WEEK

The Lazy Project Manager's week goes like this.

Monday

Monday, or the first day back after a public holiday weekend, is a long day for the Lazy Project Manager. On Monday she has three things to do.

Set targets

The first is to set targets for the week. Do this as early as possible on the first day of the new week. This not only gives the team as much time as possible to achieve their targets, but also gets people back to work mentally, that is, it brings their heads back from the weekend. (We're assuming that their bodies have already arrived!)

Use your plan, your

instrumentation to do this. Do it on a one-on-one basis or in a team meeting or a combination of the two, whichever makes the most sense. Each has advantages and disadvantages. One-on-one means everybody spends the least amount of time with you; meetings are synergistic — you catch things where dependencies exist between people. Use your common sense. Then let everyone get on with it.

Daily stuff

As we described in the previous chapter, Monday is no different from any other day and so you have to do the Lazy Project Manager's daily tasks.

Any other business

Many organizations have their own regular things that happen on Mondays: timesheets have to be filled out, regular meetings for various things take place, there are routine administrative things that have

to be done. These make up the third element of the Lazy Project Manager's Monday.

TuesdayThursday

These three days are fairly quiet for the Lazy Project Manager. All she has to do is the daily stuff (see the section [The Lazy Project Manager's day in Chapter 7](#)) and then she's finished.

Friday

Friday is busier than the preceding three days but not as busy as Monday. First there is the daily stuff to be done as before.

Then, as late as possible in the day, that is as near as possible to close of business, run round to the team, find out what happened and write a status report. These results should also be recorded in the Actual Schedule and Effort columns of the project Gantt chart or on an estimating score card (see [Appendix 5](#)). Here we are recording whether or not the

targets set on Monday were met. Those who haven't met their targets have the option of the two weekend days to make them up.

Notice a couple of things here. The first is the surgical use of overtime. Many software organizations make unlimited overtime a feature of the job. If, at your job interview, they say things like "we work hard, we play hard" or "we run aggressive schedules here," you can safely assume that they expect you not to have a life outside work. The Lazy

Project Manager has no time for such foolishness, but equally she recognizes that overtime is occasionally needed in the real world. The use of overtime by work hard play hard, aggressive schedule companies is done with all the finesse of a chainsaw operator. The Lazy Project Manager uses overtime like a surgeon with a scalpel, making fine cuts to superb effect.

The other thing to notice is that in doing this overtime, we are actually adding some contingency back into our

project. Thus, contingency is not just something you get one crack at, but a little reservoir that you can top up repeatedly. Think of it like those old arithmetic problems where a tank of water has a hole in it so that water is flowing out, but also a running tap is filling up the tank. If you can keep some water in the tank at all times, then you will never run out of contingency, and you will never have to go back to the powers that be to renegotiate a commitment.

A VARIATION ON THE LAZY PROJECT MANAGER'S WEEK

Dee Carri, a friend of mine, invented the following variation of the Lazy Project Manager's week.

Monday

As well as doing everything else on Monday, why not write the status report on Monday as

well? Dee told me about this. She writes it and pins it to the wall. She says "This is what I want to be reporting on Friday, and once I've written the report I don't like to change it. You can be sure I'll move heaven and earth rather than go back and have to edit that report."

TuesdayFriday

If you write the report on Monday, then there is nothing to do on the other four days but follow the Lazy Project

Manager's day.

Some people like to do the goal setting and the status reporting in the one meeting, but I don't think this is a good idea. It tends to blur the distinction between a target and an achievement. People say things like "I didn't quite get that done by then, but don't worry it'll be ready in the next few hours." Note that in this situation you already have a slip on your hands. By separating the two things there is no ambiguity, fuzziness or confusion about what was and

was not achieved.

APPLICATION TO SOFTWARE ENGINEERING



Nothing more to be added here
just go to it!

PSI CONTRIBUTION



The maximum PSI contribution for Step 8 is 10. Award yourself high marks here if you let all the players know what's happening on the project. Encase your project in a glasshouse and score high. Keep it all close to your chest and score low: remember what happened to Captain Scott!



STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- MAKE A LIST OF THE JOBS THAT NEED TO BE DONE**
- THERE MUST BE ONE LEADER**
- ASSIGN PEOPLE TO JOBS**
- MANAGE EXPECTATIONS, ALLOW A MARGIN FOR ERROR, HAVE A FALLBACK POSITION**

Implementing the plan; achieving the goal

6. USE AN APPROPRIATE LEADERSHIP STYLE

- **KNOW WHAT'S GOING ON**
- **TELL PEOPLE WHAT'S GOING ON**

Chapter 9. STEP 9: REPEAT STEPS 18 UNTIL STEP 10

INTRODUCTION

WHEN SHOULD WE UPDATE
THE PLAN?

APPLICATION TO
SOFTWARE ENGINEERING

PSI CONTRIBUTION

INTRODUCTION

Your job as a project leader involves repeating these first eight steps of structured project management until the project is complete. How often do you do this? Every month? Every day? Every hour? In [Chapter 12](#) I will talk about how to spread your time across projects, but basically the answer is yes! Every month, every day, every hour as often as it takes to get the job done. This means that:

- Everyone is permanently focused on the goal. The team members have heard the dream, the vision, so often that any one of them could give your speech!
- You are using your list of jobs, your plan, like a steering compass, pushing on to each new horizon. This means that you are constantly scanning the plan, adding new jobs as unexpected things arise, changing the nature of jobs as more information about

them becomes available, marking jobs off as they complete, deleting jobs if you find they are now no longer relevant and moving people to where they are most effective. At all times you are using the style most appropriate to the situation.

- You are checking to ensure that there is one and only one leader on the project.
- You are constantly thinking of the expectations you

have set and considering fallback positions. Each job has associated with it an expectation and a fallback position, and it may be possible to fall back on a number of these without affecting the overall project.

- You are out in front scouting, scanning the horizon for danger, asking "what can go wrong?" Behind you, you know what progress the team is making and you report

back to them of new perils,
obstacles, challenges
ahead.

WHEN SHOULD WE UPDATE THE PLAN?

This is a question that gets asked on almost every project management workshop that I run, and I think it's the result of a number of different issues being mixed in together and confused.

There are three different answers to the question, depending on what the questioner meant.

Constantly

As we have seen already, the plan is constantly being updated, throughout the Lazy Project Manager's day.

Weekly

These days, I would never run a project without using a PC planning tool. You may know that if you use one of these tools, then the plan (the Gantt chart) is stored in a file; and for example, in MS Project, this

file is a .MPP file. It is this file that I make the constant changes to as described above.

It is useful though, to keep copies of previous versions of the plan, so you can look back and see how things unfolded. I do this on a weekly basis. The convention I use is that last thing on Friday I copy the current plan (the current .MPP file) to a new file. This, then, becomes the one I operate on in the coming week.

I name the files using the Monday of the week in

question. Thus, XX042301.MPP would be the file for the week of Monday April 23 ('XX' is a project code) and on the Friday of that week, I would create a new file XX043001.MPP which I would then operate on in the week of Monday April 30.

Only if there's a slip

At the outset of the project we write a plan, we agree it with all and sundry, everyone gets a copy and the plan then becomes our contract with the

powers that be. This document need never be changed again unless a slip occurs which can't be recovered (see [Chapter 7](#)).

If there is never such a slip, then this particular document never changes. Status reports keep everybody posted about what's happening; tell everybody how we're progressing against this original document.

If there is an unrecoverable slip, then this means we are in a position where we must renegotiate our contract. Then

a new version of the plan document will get produced, a version 2.0 say, and this becomes the project's new contractual basis.

APPLICATION TO SOFTWARE ENGINEERING



Software projects have a notorious reputation for going out of control. We mentioned in [Chapter 1](#) the US Navy's A-12 attack plane. (By the way, while the first edition of this book was being written, the A-12 project was canceled!) One of the elements cited in the A-

12 cost overrun was software; and increasingly, as software intrudes into more and more areas of our lives, and as we use software to do increasingly complex (not to mention life-threatening!) tasks, we find that software projects go haywire.

That this should be so is hardly surprising. Software development as a discipline is less than 50 years old. When you compare it with something like the construction of buildings which has been going on for several thousand years,

it seems only to be expected that software development should still have the status of a cottage, almost a craft industry. It is one of the few labor-intensive industries remaining, and to all intents and purposes, every element of a software product is still handmade. (Even pre-processors or code generators require handcrafted input!) This is not to say that there have been no efforts to move the discipline forward. A rash of methods appeared during the 1970s and 1980s, and as a result of this it was

judged that software development had now achieved the status of an engineering discipline. Hence the phrase *software engineering*. Note that in the process, a lot of phraseology from other disciplines — for example, *sub-assemblies* (from manufacturing), *architecture*, *build* (from the construction industry) — drifted into the software engineering vocabulary.

We talked in [Chapter 2](#) about making a list of jobs to get you to the next horizon. Software

development has already defined the main horizons for you. There is the concept of a product development life cycle (PDLC) and the horizons correspond to phases in the PDLC. Anyone who has ever worked on a software project will be familiar with PDLCs. In large organizations PDLCs are an end unto themselves and departments exist to maintain them. For the purposes of this chapter, we are going to assume a software development life cycle consisting of six phases. These

six phases are:

1. A plans and requirements phase.

- High-level design. This will include the commencement of user documentation and test plans.
- Low-level design.
- An implementation phase. This will include coding, unit testing, integration, integration testing and the completion of test plans and user

documentation.

- Alpha test a test of the system, preferably by an independent team, using the test plan.
- Beta test a real-life test by a controlled group of final users of the system.

Note

Don't worry if your software development life cycle doesn't

correspond exactly to the one we are using here. Once you have seen what we do, you will see that it will fit easily on to whatever life cycle standard is used by your organization. The rest of this chapter shows structured project management operating on the project as a whole and within each of the phases.

Plans and requirements

Visualize what the goal is; set your eyes on the prize

The first step on all software projects is (or should be) a statement of requirements; called in our PDLC a requirements specification (RS). The RS gives the requirements that the software system to be developed will satisfy. Thus it describes:

- what the system will do
- how it interacts with the world outside it
- the performance levels expected of it

Properly written, the RS is a statement of the prize. There are various ways you can approach writing a RS.

- At its most basic, the above three items constitute a checklist for a RS.

- There are also numerous checklists in the literature; as well as an increasing number of methodologies for identifying, documenting and tracking requirements. All of these fancy tools have their place and can help, particularly in controlling the vast amount of information which results from a system development effort.
- My company, ETP, offers a consultancy service called AAD (accelerated analysis

and design) which is basically a 5 – 10 day brainstorming session in which all of the requirements and some of the design of a system can be carried out, provided that all the decision-makers on a project are involved.

Also, irrespective of what checklists or tools you may be using, you should still go through the visualization described in [Chapter 1](#).

Make a list of the jobs that need to be done

Treating the RS as the first horizon very quickly enables us to identify the jobs we have to do to get there. This list is very straightforward, and is probably something like:

- get the RS written
- one or more levels of review (internal/external)

- one or more levels of rework
- sign-off

Visualizing the prize enables us, to some extent, to see beyond that first horizon and build our job list. We know what the next big horizons are because our PDLC gives them to us. Also the last few jobs to be done (put the software on to a disk, ship it, install it at the user site, etc.) are very apparent. The bit in the middle will be a trackless waste of

pretty much Polar proportions, and all you can do at this stage is to estimate it. Estimation is discussed in detail in [Chapter 14](#).

There must be one leader

There is nothing more to say here over and above what was already said in [Chapter 3](#).

High-level design

We have reached the first horizon: the RS is written and, more importantly, we can picture the goal in shining detail. However, our immediate interest now focuses on getting a high-level design (HLD) done. If the RS describes what the system will do, what it looks like on the outside, the HLD describes what the major bits of the system's internals look like — the heart, lungs, liver, and so on.

Visualize what the

goal is; set your eyes on the prize

The HLD can be pictured as a block diagram showing your system's major components. Eventually this block diagram will decompose into many smaller blocks, right down to the program or module level, but for the moment it just has the big blocks.

Make a list of the jobs that need to be done

A cycle similar to that for the RS; that is, write, review, rework, sign-off is probably all that is required here.

Low-level design

Second horizon reached. We must now decompose the HLD down to a sufficient level of detail that the system can be coded.

Visualize what the goal is; set your eyes

on the prize

In our PDLC we have done our design in two jumps, HLD and LLD (low-level design).

Irrespective of how many jumps you take to get there, and what you call these jumps, you must end up with a picture of your system. The picture is a block diagram showing all the components of the system to be built, right down to the smallest one. For large systems this may occupy many pages and have its own internal

hierarchy. This now becomes the prize.

Make a list of the jobs that need to be done

The familiar cycle of write, review, rework, sign-off is what you need. I try to make it a rule never to commit to delivering anything until the LLD is complete. Most customers are happy with this, or if they insist on a fixed-price, cast-in-concrete estimate

sooner than that, you make it clear to them that they are going to pay a heavily-loaded price, so that you can allow for your own risk and exposure (we discuss this problem again in [Appendix 1](#) in the section [Build in contingency](#)).

Your job list will be a living thing, constantly changing to adapt to circumstances. New jobs will appear, jobs will complete, jobs will be found to be unnecessary and jobs will end up differently from what you originally envisaged. (This is why a computerized project

planning system is very useful.) All these changes are good, routine stuff. Your list will be highly detailed for the short horizon and less so for the longer one.

Tell people what's going on

The only new point to be made here is this: if, during the design process, something emerges which significantly modifies expectations, then it

needs to be made public. Good surprises you can keep secret until the moment you feel is best for you and your purposes. Bad ones, unfortunately, have to be made public, so that the outside world can adapt to the new situation.

Implementation

Design is complete and so we move on to writing and testing the code.

Visualize what the goal is; set your eyes on the prize

The goal is now a working system. If you are building your system as a series of increments, each having gradually more functionality than the previous one always a good way to proceed because of the fallbacks it allows you then this phase will contain a number of mini-phases each of which will represent an horizon along the

way.

Alpha test

Your system is complete and works to the developer's satisfaction. The test plan is now run against the system, preferably by somebody other than the developers.

Visualize what the goal is; set your eyes on the prize

The goal is a working, bug-free (insofar as is humanly possible) system or product.

Beta test

Your system or product is released to a limited number of users/customers. It operates under field conditions and corrections/enhancements are made to it by the developers.

Visualize what the goal is; set your eyes

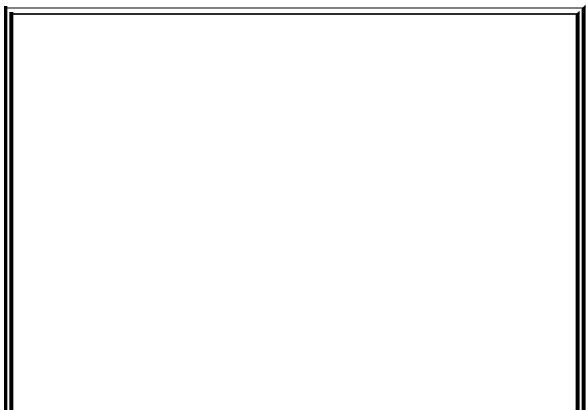
on the prize

The goal is what you labored for all this time: a system or product that the users love and/or that sells in thousands.

PSI CONTRIBUTION



Step 9 scores 0 because the score is contained in Steps 1 - 8.



STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- MAKE A LIST OF THE JOBS THAT NEED TO BE DONE**
- THERE MUST BE ONE LEADER**
- ASSIGN PEOPLE TO JOBS**
- MANAGE EXPECTATIONS, ALLOW A MARGIN FOR ERROR, HAVE A FALLBACK POSITION**

Implementing the plan; achieving the goal

6. USE AN APPROPRIATE LEADERSHIP STYLE

- **KNOW WHAT'S GOING ON**
- **TELL PEOPLE WHAT'S GOING ON**
- **REPEAT STEPS 1 - 8 UNTIL STEP 10**

Chapter 10. STEP 10: THE PRIZE

"What is now achieved
was once only imagined."

William Blake

THE PRIZE

Finally there comes the day you've dreamed of, the day that was at the heart of your vision, the day when the project is completed. Amundsen and his companions arrived at the South Pole at 3:00 in the afternoon of Friday, December 15th, 1911. They recorded different feelings in their diaries. Bjaaland, the ski-champion, was delighted to be there before Scott, and celebrated with a big meal.

Hanssen, the dog-driver, was just glad that the wind wouldn't be blowing into his face any more, but would now be at his back.

Trevor Griffiths, in his screenplay for the television series *The Last Place on Earth* ([Griffiths, 1986](#)), about Scott and Amundsen, puts the following words into Amundsen's mouth when he is asked by his companions to make a little speech.

I thank you. From my

heart. For your work. Your
commitment. Your craft
and your comradeship. I
have ... no grand
emotions, no profound
thoughts to share with
you, I have to confess it. I
experience the
excitement, of course ...
But above all what I feel is
... not large or deep at all,
it's just: how good it is. To
be alive.

THE RECKONING

We humans have short memories; and there is nothing more easily forgotten than the experiences learned on a project. If it was a bad project we want to blot out those awful memories, while if it was a good one, we are euphoric and just want to take on something bigger and better. Pause just for a short while. Look back over a project and ask yourself questions like the following:

- Did you end up exactly where you said you would, or was the goal achieved different from the goal predicted? Where did it change? Were you aware that it had?

Document how your estimates turned out against what actually happened:

- What did you do right and wrong, particularly with regard to people? What mistakes did you make? Were there things that just

couldn't have been done better? If you could do it over again, what would you do differently?

- What surprises occurred? What things did you not anticipate?
- Did you have to eat into your margin for error? Did you have problems with people's expectations? Did you have to actually go to your fallback position? What lessons do these hold for the future?

Write down the answers to these and questions like them, and have them there beside you when you set out to plan the next project.

PSI THRESHOLDS

Since the first edition of this book appeared, we have found out essentially three things about the PSI, as discussed in the following three sections.

Threshold for Steps 15

A score of 40 gained from Steps 1–5 appears to be a threshold. Initially, when a project starts it will score little, if anything, on the PSI scale.

We expect this to happen.

Your priorities on the project then are to firm up the goal, what is to be delivered (Step 1), and the plan for delivering it (Steps 2–5). You do this by going through those early project phases: requirements definition, analysis, various levels of design. If these are done properly, they will have the effect of raising the Step 1 score and this will in turn enable you to increase the scores from the other four steps. The score should eventually go above 40 as

these various phases are completed.

If you cannot register a score above 40 it means that the goal is not sufficiently well worked out and agreed. *Unless you take steps to firm up the goal in as much detail as possible, you should not go any further in the project.* You should certainly not begin implementation-type activities. *Your project will not succeed.*

Threshold for Steps 110

A score of 60 gained from Steps 1 - 10 appears to be a threshold. Initially, when a project starts it will score little, if anything, on the PSI scale. We expect this to happen.

From then on, the low scores will always point you at your most important tasks. If you are a person who has difficulty prioritizing, then the low scores on each step will show you what your absolutely most important jobs are. If, at any given time, you are focusing on these, you are doing exactly the right things.

The Second Law of Project Management

[Brooks \(1987\)](#) cites Brook's Law: "Adding people to a late project makes it later." Note that this law involves two of our four parameters: delivery date and effort.

The relative balance of the two PSI thresholds described above implies that a much more general version of this law exists. I have called this law The Second Law of Project Management. It works like this:

- If you got a very high score out of Steps 6 – 10, say you got 9 out of 10 for each step, a total score of 27, this would imply an extraordinary level of sensitivity in handling staff (Step 6), monitoring and control (Step 7) and reporting (Step 8).
- Now, if you get a low score out of Steps 1 – 5, a score that is under the 40 threshold, then the extraordinary score from Steps 6 – 10, added to the

low score from Steps 1 - 5, won't be enough to bring you over the 60 threshold.

Or to put it more bluntly:

Doing anything you like on a poorly planned project won't make the blindest bit of difference.

The Second Law of Project Management involves all four of our parameters and is, I believe, a more general law, of which Brook's Law is a particular instance or occurrence.

APPLICATION TO SOFTWARE ENGINEERING



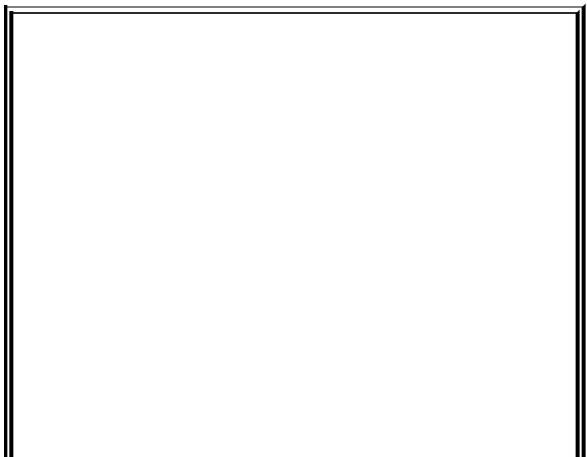
The last act of the drama is the achievement of the final horizon: the Prize. When it comes it all seems so familiar, and it should be, because it is a daydream in which you have lived ever since you first set out on the road. It's time for you to celebrate in whatever

way you had planned to. With luck, they'll give you time to do an audit. (Insist on it, even if they don't!) After that they'll probably put you on a new project — even more challenging, even more exciting, with even more outrageous deadlines and delivery dates. But that is tomorrow. Today, you can bask in your achievement, and marvel at how easy you made it look.

PSI CONTRIBUTION



Step 10 is 0. It's academic now
you're finished.



STRUCTURED PROJECT MANAGEMENT

Planning the project

1. VISUALIZE WHAT THE GOAL IS; SET YOUR EYES ON THE PRIZE

- MAKE A LIST OF THE JOBS THAT NEED TO BE DONE**
- THERE MUST BE ONE LEADER**
- ASSIGN PEOPLE TO JOBS**
- MANAGE EXPECTATIONS, ALLOW A MARGIN FOR ERROR, HAVE A FALLBACK POSITION**

Implementing the plan; achieving the goal

6. USE AN APPROPRIATE LEADERSHIP STYLE

- **KNOW WHAT'S GOING ON**
- **TELL PEOPLE WHAT'S GOING ON**
- **REPEAT STEPS 1 - 8 UNTIL STEP 10**
- **THE PRIZE**

Part 3: RUNNING MULTIPLE PROJECTS SIMULTANEOUSLY

Chapter 11. THE LAZY PROJECT MANAGER'S MONTHLY ROUTINE

INTRODUCTION

PROJECT MANAGER'S
MONTHLY ROUTINE

INTRODUCTION

In the introduction to [Part 2](#), we put forward the (rather glaringly obvious) argument that if we could bite off the amount of a project that our plan required of us every day, every week and every month, then our project would remain on target. The same applied to the project budget. Clearly the same argument applies to multiple projects. If the project manager can succeed in meeting each of his projects'

successive monthly, weekly and, ultimately, daily cost and schedule milestones, then the projects will remain on schedule.

The procedures described in this and the following two chapters enable the project manager to do exactly that.

These three procedures *must* be used in conjunction with each other in the way described here. An individual procedure cannot be used in isolation.

PROJECT MANAGER'S MONTHLY ROUTINE

If a project manager has planned all of his projects using the Ten Steps, then each project will consist of a plan which is very detailed to the next milestone, probably no more than 4 – 6 weeks away, and as detailed as it can be thereafter.

The project manager's monthly routine involves the project manager taking a month-long

bite out of each project and then trying to organize his time such that all the tasks in this month-long bite are carried out.

To do this, go through the following steps either at the beginning of a new month or the end of the preceding one.

1. For each project identify where the project has to be by the end of the month.

- Identify how much effort is involved *by you personally* (not

your team) in each of these projects. Include in your tally, time that will be spent on things like:

- trips
- pre-arranged meetings
- training
- annual leave
- public holidays
- personal tasks

- quality issues
- support
- miscellaneous tasks, e.g. dealing with the in-tray, interruptions, general administrative things that can't be tied to any particular project

Everything you do in your job needs to be accounted for in this tally.

- Add up this effort required by all the projects and tasks,

and convert the resulting figure to man-days.

- Compare this against the number of working days available.
- If (3) is less than or equal to (4) then proceed to (6). Otherwise, you need to determine how the shortfall will be made up. Basically, there are four choices open to you:
 - do less delegate it
 - do less drop it

- work more hours
- delay things and extend deadlines

Note that the latter may have the effect of delaying projects. The demand, from (3), needs to be less than or equal to the supply, from step (4), before you can proceed. Once this is true you can proceed to (6).

- Now schedule everything on a personal Gantt chart as shown in [Figure 11.1](#). This can be done on paper, but a

spreadsheet is very useful for doing this, and makes changes from month to month much easier.

Figure 11.1. A personal Gantt chart

						Total	Actual
August						days	days
	1st-5th	8th-12th	15th-19th	22nd-26th	29th-31st	21	21
New system							
Installation process		3				3	4
Integration							
Meeting	0.5					0.5	2
Plan	2					2	4
Pick supplier				3		3	0.5
DBA							
Install client upgrade					2	2	0
Routine work	0.5	0.5	0.5	0.5	0.5	2.5	0.75
Admin							
Management	0.5	2	1.5			4	1.5
General admin	0.5		0.5	0.5	0.5	2	3
Assistance	0.5	0.5	0.5	0.5	0.5	2.5	8
Interruptions	0.5	0.5	0.5	0.5	0.5	2.5	2
Total	5	6.5	3.5	5	4	24	25.75

- Now, when changes occur in the course of a month or new requests/ demands are made on the project manager's time, he can schedule these into the overall monthly view using the options described in (5) above, rather than making unreasonable or unachievable commitments.

Chapter 12.

PROJECT

MANAGER'S

WEEKLY ROUTINE

[INTRODUCTION](#)

[SPECIFIC WEEKLY JOBS](#)

INTRODUCTION

The monthly view shows what jobs have to be done each week for the project manager's projects to stay on schedule. The weekly routine involves extracting from the monthly view a week's worth of work. This should be done at the end of a particular week or at the very beginning of the next week.

To build a weekly schedule, the project manager reads from the

monthly view the contents of the coming week, and schedules this over the five (or six or seven, but ideally five) days of the week. Let's illustrate this by taking a week from the example shown in [Figure 11.1](#).

Week of 15 August

The project manager will spend about half the week involved in routine work, but there is one big job, the integration plan, that is going to require two

days' solid work. If this is the case, then the project manager might set two specific days aside for this, e.g. Tuesday and Thursday. On these days he would be saying that no routine work will be done, and anyone requiring these things will have to wait. Thus his week would look like:

Monday	Routine stuff
Tuesday	Start integration plan
Wednesday	Routine stuff
Thursday	Finish integration plan

Friday

Routine stuff

Again, if changes occur in the course of the week, or new priorities arise, the project manager can reschedule across the week, and if necessary, into subsequent weeks, using the monthly view as an overall context within which to make decisions.

SPECIFIC WEEKLY JOBS

There are two specific jobs that should form part of every project manager's week. These are Monday morning project planning meetings and producing a project status report last thing on Friday. These were described in detail in [Chapters 7](#) and [8](#) respectively. We summarize them again here.

Monday morning

planning meetings

Planning meetings should be held as close to the beginning of the week as possible. Their aim is to agree what proportion of the project schedule each person is going to bite off that week. Delays, people absences and unforeseen things occurring on projects can all be dealt with at this meeting. The project manager can use the PC-based model of his project to make sensible decisions about how to deal with unexpected events.

Friday project status report

The status report should be done last thing on Friday and the report itself, or versions of it, delivered to the customer, management and the project team. Where the Monday project meeting sets targets, the Friday status report records results.

These results should also be recorded in the *actual schedule and effort columns* of the project Gantt chart or on an

estimating score card.

Chapter 13.

PROJECT

MANAGER'S DAILY

ROUTINE

INTRODUCTION

INTRODUCTION

Ultimately, whether projects stay on schedule or not will be determined by what the project manager does or doesn't do on a daily basis. The daily routine tries to ensure that those jobs which are key to the success of each project get done. The daily routine involves choosing those jobs which are going to get done every day. This should be done first thing in the morning or last thing on the preceding day.

1. Draw up a list of all the jobs that are contenders to be done on the particular day. To do this consider the following sources:

1. Projects will have generated jobs through the monthly and weekly routines already described.

2. Projects will have generated jobs through the project manager looking daily at the plan for each project under

**her control, and
determining which jobs
require some action to
be taken.**

- Projects will have generated jobs through the project manager trying to generate more detailed task lists about parts of the project which still lie in the future and which have not yet been reduced to a man day level of detail.
- There will be other jobs in-tray, meetings, faxes, e-mails, mail, yellow stickers,

reminders, phone calls to be made.

- There may also be personal jobs.
- Having developed the list of jobs, categorize them according to the following scheme:

1. *Have to get this done today*

- Nice to get this done today
- Not going to get this done today

- Delegate this job
- Now, do all the *Ds*, and all the *As*. Everything else can be moved to later in the week or month.

Note

Note that this approach only works where you have first done the monthly and weekly analyses. If you don't do these, then the only result will be that a great

backlog of tasks will build up for later in the week or month.

As described above, if changes occur in the course of the day, or new priorities arise, the project manager can reschedule simply by categorizing the jobs again, and doing the *As* and *Ds* as before. If *A* jobs end up not getting done, then, by definition, they couldn't have been *A*

(*have* to get done)
jobs to begin with.

Part 4: HOW TO ASSESS PROJECT PLANS

Chapter 14.

ASSESSING

PROJECT PLANS

INTRODUCTION

FIRST LEVEL CHECKS

SECOND LEVEL CHECKS

THIRD LEVEL CHECKS

GUIDELINES FOR WRITING
PROJECT PLANS

INTRODUCTION

You will have gathered by now what our angle in all of this is: if you plan it right, then the doing will be very straightforward. The key measure of how well a project has been planned is the quality of the project plan that has been produced. In this chapter I will show you how to assess such plans — your own or other people's. By learning how to do this you will stop potential disasters from becoming real

ones.

As you also know by now, I am anxious to make you into not just a successful project manager, but a Lazy one. The method I will show you in this chapter will be a key weapon in your Lazy Project Manager's arsenal. If the plan is a poor one you will find out within a few minutes. Then, only if the plan is good will you spend time analyzing it more closely. The tests contained in the method described will smoke out the turkeys with the least amount of effort on your part.

As you would expect, the tests in the method are directly derived from the Ten Steps.

At the highest level there are four things you need to look for in assessing a project plan. These are:

1. That the project has a clearly defined goal.

- That there is a plan for achieving the goal.
- That the project has a leader.

- That there is some sort of backup plan or room for maneuver.

All of the checks I describe will be to determine the existence or otherwise of these.

It may be convenient for you not to do all of the checks described on every plan you encounter. In particular, it might be useful for you to have a first battery of checks that a plan must pass, and only if it passes these would you analyze it further. Ideally, too, the

amount of work involved in carrying out this first battery of checks wouldn't be all that great.

This is exactly the way I have organized the checks in this method. In all, there are nine checks, and these are listed in the following table. Opposite each check is an assessment small, medium, large of the amount of work required to carry out the check.

Five of the checks require relatively minor amounts of work. I suggest you use these

as your first level of analysis. The five of these could probably be done in under 30 minutes.

Another three of the checks can be done with modest amounts of work. To do them all might take up a couple of hours of your time. However, as I have said already, you would only do them provided the plan being tested had already passed the 30-minute test.

The remaining check could, depending on the scale of the plan, turn out to be very time-consuming indeed. Whether or

not you would do such check is, of course, entirely up to you. If you personally decided not to, it would still be important that somebody and this is most likely to be the project manager should do it. The checks we will consider are these:

CHECK

EFFORT INVOLVED

First level checks

Contents

Small

WBS

Small

Gantt	overview	Small
-------	----------	-------

Resource checks 1 and 2		Small
-------------------------	--	-------

PSI		Small
-----	--	-------

Second level checks

Schedule and effort		Medium
---------------------	--	--------

Gantt critical path		Medium
---------------------	--	--------

Resource loading checks 3 and 4		Medium
---------------------------------	--	--------

Third level checks

Gantt	all jobs	Large
-------	----------	-------

FIRST LEVEL CHECKS

Project plan: contents analysis

The first quick and very simple check you can do on a project plan is to check its table of contents. (If it doesn't have one send it back!) This is likely to take no more than 5 – 10 minutes and can tell you whether it is worthwhile you spending any more time than this assessing the plan.

They may have different names but the table of contents should contain the following elements:

1. *Management summary.*
Some sort of overview or summary.

- *Statement of what is to be done.* The goal of the project.
- *Deliverables.* Also part of the goal. These may come under a number of headings:

1. Software

- Hardware
- Documentation
- Services
- *Completion criteria.* How do we know when the project is over? (A key part of the goal.)
- *Acceptance criteria.* How does our customer decide the project is over? (Also a key part of the goal.)

The next eight are all part of the plan and the backup plan.

6. *Work breakdown structure (WBS).* A list of all the jobs that have to be done to complete the project together with estimates of their associated effort and any assumptions made.

- *A Gantt chart.* Showing the WBS phased over time, and stating any assumptions made.
- *Milestones.* A list of key dates extracted from the Gantt chart.

- *Resources required.* Human and otherwise.
- *Resource loading.* How the resources are phased over time.
- *Project budget (optional).* Derived from the effort in the WBS (6) and Resources (9 and 10).
- *Project organization chart (optional).* Among other things, this serves to identify the project leader.

- Backup plan/margin for error. Sometimes called risk analysis.

The project plan represents a prediction of how things will go in the future with regard to this project. The intention of this section is to show that the author of the plan has given some thought to what he will do when what happens in reality differs from the project plan.

Match these 13 elements against the table of contents of

the plan you are assessing. Without delving too deeply, this will give you your first indication of how well put together the plan is.

Project plan: work breakdown structure analysis

The next thing you need to see is whether or not the plan is complete, in the sense that it contains all of the various work elements that make up the project. The checklist displayed

in Example 1 allows you to do this. Use it as a guide to highlight missing Project Plan elements. The checklist is presented in the form of a WBS, a structure that breaks down all of the project elements from a high to a detailed level.

The WBS tries to make the most general possible assumption about the life cycle that the project follows. The life cycle can be described as follows.

1 The requirements for the system are developed.

2 Following on from this, an overall architecture for the system is developed. This architecture gives the major blocks of the system.

3 The system is programmed (3.1) and a working (integrated) version (3.2) of it is produced.

4 Testing is planned (4.1)

and carried out (4.2). 4b is often referred to as alpha test.

5,6 Both while the system is under development and after it is released, accepted, and has gone into maintenance, elements 5 and 6 (project management and configuration management) operate on the system.

7 Manuals are developed.

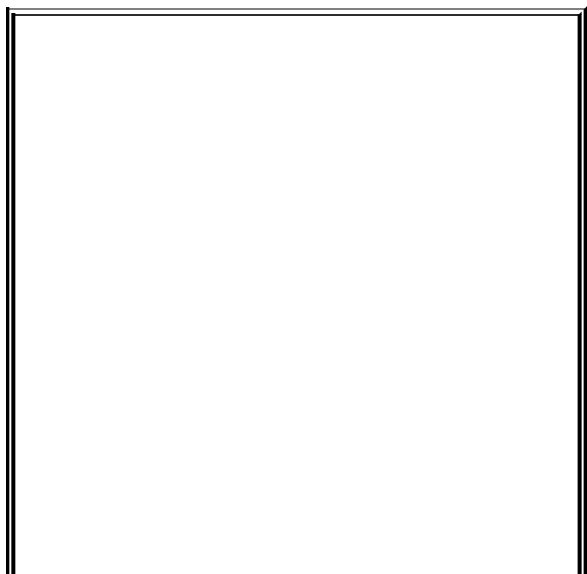
8 The system is released and implemented and accepted.

9 Training, initially of project team personnel and eventually of system users, takes place.

10 The system, having been accepted, goes into maintenance.

11-13 Some other elements that are likely to occur on almost any project.

The WBS is presented in such a way that you can add in your own WBS checklist elements, gathered from your own experiences over time.

A large, empty rectangular box with a double-line black border, occupying the lower half of the page. It is intended for the user to add their own WBS checklist elements.

EXAMPLE 1: WORK BREAKDOWN STRUCTURE

1. Requirements analysis

1.0 Reviews

1.1 Requirements document(s)

1.2 Request for proposal

● Product design

2.0 Reviews

2.1 High-level design
(basic architecture)

2.2 Low-level design
(detailed design/program
specs)

2.3 Possible development
of prototypes

- **Programming**

3.0 Code walkthroughs

3.1 Code and unit test

3.1.1 Write code
element

3.1.2 Clean compile

code element

3.1.3 Unit test code element (including element test plan)

3.1.4 Code element documentation

3.2 Integration

3.3 Test environment development

3.4 Development support

3.4.1 Database

administration

3.4.2 Development
environment

3.4.3 System build

- **Testing**

4.0 Reviews

4.1 Write test plans

4.2 Alpha test (system
test)

4.3 Beta test (acceptance

test)

- **Project management**

5.0 Reviews

5.1 Production of initial project plan and ongoing project monitoring and control

5.2 Management of subcontractors

- **Configuration management**

6.0 Reviews

6.1 Ongoing configuration management

6.2 Release

- **Documentation**

7.0 Reviews

7.1 User (different types)

7.2 Administrator

7.3 Release notes

7.4 Technical manuals, i.e. manuals describing how the software works

7.5 Help texts

- **Implementation**

8.0 Reviews

8.1 Installation

8.1.1 Plans

8.1.2 Activities

8.1.3 Test

8.2 Conversion

8.2.1 Plans

8.2.2 Activities

8.2.3 Test

- **Training**

9.0 Reviews

9.1 Familiarization by project personnel

9.2 Training of project personnel

9.3 User training

- **Maintenance**

- **Staffing**

11.1 Recruitment

11.2 Staff training

- **Quality**

12.1 Quality management

12.2 Quality plans

- **Administration**

13.1 Public holidays

13.2 Annual holidays

13.3 Sick

13.4 Meetings

Match these WBS elements against those in the plan you are assessing. This will show you what, if anything, has been forgotten in your plan.

Project plan: Gantt chart analysis, high-level overview

The Gantt chart is the heart of your project plan. If there isn't one in the plan you are assessing, then send it back now, because there's not much point in going any further.

The Gantt chart can be assessed at a number of different levels, each one becoming progressively more detailed than the previous one.

The more detailed the assessment the more likely you are to catch potential flaws in the plan. However, more work is required on your part to do this detailed analysis.

The three levels we will look at, working from the highest to the lowest, are:

1. High-level overview

- Critical path
- All jobs

At this stage, however, we will

only concern ourselves with a high-level overview.

The plan should present an overall high-level overview of the project. In other words, the plan should show:

- What the main phases of the project are
- How long these phases last
- How they relate to each other
- The amount of work in each

phase

It doesn't take long to check this, and there are a number of advantages to doing so.

First, this overview gives you milestones — points in time at which the project has to be in a particular state. If, when a particular milestone comes, a project isn't at the expected stage, then this is early warning that something is wrong.

The presence of a high-level overview also implies clarity of

vision on the part of the project manager. It indicates that the project manager has clearly thought through at least at a high level what he is being asked to do.

An example of a plan which has exactly the kind of overview we are looking for is shown in [Figure 14.1](#). [Figure 14.2](#) shows one which is almost the direct opposite of what we mean. Both plans are based on real life.

Figure 14.1. Plan with high-level overview

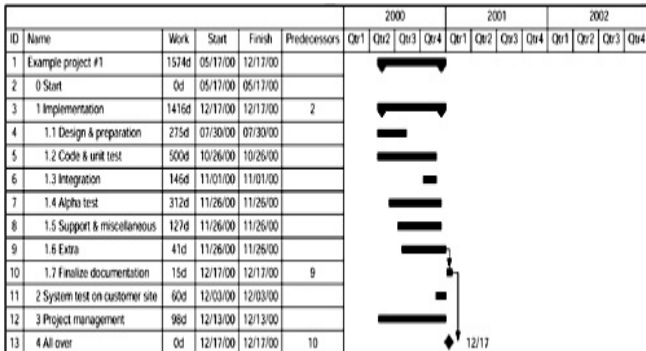
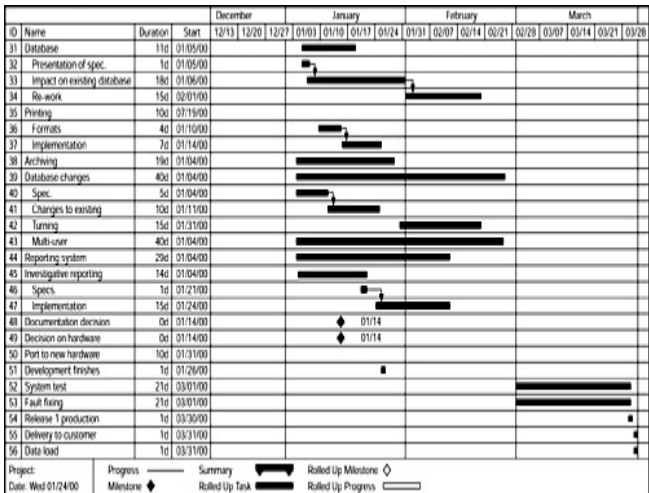


Figure 14.2. Plan with no proper overview

graphics/14fig02a.gif



One other thing to watch for at this level is the amount of activities being paralleled. Sometimes if a project has a very tight deadline or a deadline which has been set in

advance of the plan being drawn up, the way people try to allow for this is to run several activities in parallel. There is nothing intrinsically wrong with this unless the activities that are being run in parallel are actually ones which in fact cannot be. For example, a number of potential system suppliers can't really be assessed until a Request For Proposal (RFP) has been drawn up. (An RFP is a document sent out by a company inviting other companies to tender for certain products or services.) It

may be possible for some of the assessment jobs to be done, for example gathering information about particular products or systems or suppliers, while the RFP is still being drawn up, but it is certainly not possible for the RFP and the supplier assessment to complete at the same time!

Project plan: resource analyses

Two quick checks that should be done on a project are to

examine its resourcing. At its simplest what this means is that the scheduling of people on the project takes account of some basic realities.

The first of these realities (Check 1) is that the allocation of people to jobs takes into account the fact that people can only work five days a week, and that there are certain holiday periods when they don't work. Thus you should check that the effects of:

- annual leave (especially for

projects running over the summer)

- public holidays
- long weekends
- seasonal holidays
(Christmas, Easter)

have been accounted for.

Also, projects or phases of projects which end on Saturdays and Sundays should be looked at with some suspicion. They can imply that

some basic thinking things through has not been done.

Check 2 is to check that every job in the project plan has a human being's name against it (see [Chapter 4](#)). If this is not the case, then that is not necessarily a problem provided there are jobs elsewhere in the plan e.g. hiring, recruitment, internal transfers, agreeing of subcontracts, and so on to supply these human beings' names.

Project plan: PSI analysis

If you've read [Chapters 1 10](#), you'll know what the PSI (probability of success indicator) is. If not, have a quick scan through [Appendix 3](#) which will give you an overview.

The PSI is quickly calculated (you can use [Figure 14.3](#) to help you) and will not only show you what kind of shape a project is in, but the low scores will enable you to say, in your review of the project plan, where the project manager should focus his energies.

Figure 14.3.

The Ten Steps

Planning the project

Weight

- | | | |
|--|----|----------------------|
| 1. Visualize what the goal is; set your eyes on the prize | 20 | <input type="text"/> |
| 2. Make a list of the jobs to be done | 20 | <input type="text"/> |
| 3. There must be one leader | 10 | <input type="text"/> |
| 4. Assign people to jobs | 10 | <input type="text"/> |
| 5. Manage expectations, allow a margin for error, have a fallback position | 10 | <input type="text"/> |

Implementing the plan; achieving the goal

- | | | |
|--|----|--|
| 6. Use an appropriate leadership style | 10 | |
| 7. Know what's going on | 10 | |
| 8. Tell people what's going on | 10 | |
| 9. Repeat steps 1–8 until Step 10 | | |
| 10. The prize | | |

Total

100 ☐

Current projects

1-5

5-10

[illegible]

SECOND LEVEL CHECKS

Project plan: schedule and effort analysis

A very effective way of finding out if a project has been planned out and thought through properly is to examine the distribution of effort and schedule over the life of the project. By comparing this distribution against that of previous completed projects, you can get a feeling for the

accuracy or otherwise of the estimates in the plan being reviewed.

Obviously projects vary widely, from organization to organization and within organizations. Thus, the distribution of effort and schedule over the life of the project will also vary. However, finding that the distribution of the project being assessed is roughly comparable to that of a previously completed successful project will add to your confidence in the project plan being assessed. Conversely,

finding a wide discrepancy between the two distributions will cause you to ask questions about the various estimates and how they were arrived at.

Obviously you can only carry out this analysis if you already have some distribution figures that you can use as a basis for comparison. Unfortunately, it is still something of a rarity in the IT industry to find this done with any degree of consistency. Thus, to make the most effective use of this test will require you to start recording distribution data on your

completed projects. A form which makes it easy for you to do this, called an estimating score card (ESC), is described in [Appendix 5](#).

In the absence of data of your own, you can still apply the analysis by using data supplied on the ESC shown in [Figure 14.4](#). The comparison will not be as accurate, because the data are, by necessity, vanilla data culled from many different projects. However, they will enable you to get started with this analysis, and can be used until data of your own becomes

available. The points to note are:

Figure 14.4.

Estimating score card

Project Schedule & effort analysis - example

Author

Revision No.

Date

Phase	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%
Regs & design (1, 2)					27		18					
Code & test (3a)					37		36					
Int (3b)					18		9					
System test (4)					18		20					
Proj mgmt (5)					-		6					
Doc & support & CM (6, 7, 9)					-		11					
8, 10-					-		-					

- The ESC covers all but two phases of the life cycle described in the section on [WBS analysis](#). These are 8 *Release Implementation and Acceptance* and 10 *Maintenance*.
- The first column of figures shows what proportion of the elapsed time of the project should go into each of the phases. Thus, for example, 26 percent of the

elapsed time of the project would go into the requirements and design phases of a project.

- The second column of figures shows what proportion of the project's effort should go into each of the project's phases.

Note

Just to repeat again that these are general-purpose figures and so should

be treated with caution. Figures on the project being assessed may vary considerably from these, and could still be right because of the peculiar nature of that project.

Now, let's see a sample analysis. Let's say that the project plan that you are assessing contains the

distribution of effort and
schedule shown in [Figure 14.5](#).

Figure 14.5.

Estimating score card

Project Schedule & effort analysis - example

Author

Revision No.

Date

Phase	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%
Regs & design		17		7								
Code & test		55		52								
Integration		0		0								
System test (4)		28		35								
Proj mgmt		-										
Doc & support & CM		-		6								

If the author of the plan hasn't presented such a distribution, there are two choices open to you: either send it back and ask for it, or work it out yourself from the estimates of effort and schedule on the project plan's WBS and Gantt chart.

[Figure 14.6](#) shows the distribution from the project plan compared against our general-purpose figures. *Significant* differences between

the two sets of figures are discussed below.

Figure 14.6.

Estimating score card

Project Schedule & effort analysis - example

Author

Revision No.

Date

Phase	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%
Regs & design		17		7	A	26		18				
Code & test		55		52	B	36		36				
Integration		0		0	C	18		9				
System test		28		35	D	18		20				
Proj mgmt		-										
Doc & support & CM		-		6	E	-		17				

1. There doesn't appear to be enough time or effort allowed in the project being assessed for requirements gathering and design.

- Too much of the project elapsed time and effort is being taken up by the actual writing of the software.
- Integration hasn't been allowed for at all.

- System test looks too high. Is this because not enough design will have been done and so lots of problems are expected?

These are questions to which you will have to get answers before you can give this project the go-ahead.

Project plan: Gantt chart analysis, critical path

As we mentioned earlier, the Gantt chart can be assessed at

a number of different levels, each one becoming progressively more detailed than the previous one. The more detailed the assessment the more likely you are to catch potential flaws in the plan. However, more work is required on your part to do this detailed analysis. The level we will look at in this check is the critical path level.

The critical path is the shortest path through your project: in other words, it is the shortest time in which the project can complete. If any of the jobs on

the critical path are delayed then the project will be delayed.

If you want to delve deeper into the project, without actually going through it job by job, then the critical path is both a convenient and a crucial way of doing exactly that.

To do this, the plan you are assessing should show clearly what the critical path is. PC-based project planning tools do this automatically. If the Gantt chart hasn't been done using such a tool, then the author of

the plan needs to have worked out the critical path manually. If the plan doesn't show the critical path then you would be quite justified in sending it back, as the implication would be that the project manager hasn't thought it through.

Assuming that the Gantt chart does show the critical path, then for each job on the critical path you should check all of the following:

- Does the amount of effort associated with the

particular job seem reasonable?

- Do the calendar days on which the job is due to be carried out make sense? For instance, have tasks been scheduled on public holidays or are unrealistic assumptions being made about what will get done over a holiday season like Christmas?
- Does each job on the critical path have somebody's name against it

not the name of an organization, but the person who will cause the work to be done?

The same comments made earlier about paralleling of tasks also apply here. Are tasks which actually have a dependency between them being run in parallel?

All of these tests applied to each task on the critical path will generate review comments of the plan.

Finally, if you wanted to, you

could do a simple risk analysis using the four parameters from the First Law of Project Management:

- functionality what is to be done
- delivery date (or elapsed time) when the job must be done
- effort how much work is involved
- quality how well it is done

You would do this by asking yourself questions like:

- Is this job doing something which is crucial to the project? If not, could it be put into a later delivery? (Functionality)
- What happens if the job runs over? (Elapsed time)
- Can the job be shortened? (Elapsed time)
- What happens if the job is bigger than we estimated?

(Effort)

- What would be the effect of adding more people?
(Effort)
- What happens if this job isn't done perfectly?
(Quality)

Project plan: resource loading analyses

These are two other checks to see whether the resource loading has been done

correctly, and that the project takes account of some basic realities.

Check 3 is to see that people have not been scheduled more than 100 percent, that is, they are not scheduled to be in two places at once or working more than five days per week.

Check 4 examines whether the amount of people's effort that has been scheduled onto a particular project takes into account their other commitments. Thus if a person has been scheduled five days

per week, then it means that *they will be doing absolutely nothing else* during their time on the project. Often people have several commitments and these have to be taken into account.

It should be said again that the use of a PC-based tool by the project manager makes checking all of these things very straightforward. In particular, a PC-based tool can supply resource loading charts showing how much work each person is scheduled to do, and when.

THIRD LEVEL CHECKS

Project plan: Gantt chart analysis, all jobs

As we mentioned earlier, the Gantt chart can be assessed at a number of different levels. The level we will look at in this check is that of all the jobs in the plan.

The Gantt chart analysis can be applied to all jobs in the project. This is something that

could be quite time-consuming. While you might expect the project manager to do it, you probably wouldn't do it in a plan you were assessing. What you might do, however, would be to do it in the form of spot checks on particular jobs. For instance, you might spot check:

- particularly large jobs
- jobs to be done by an outside contractor
- jobs to be done by somebody new to an

organization

- jobs that are known to be technically complex
- jobs that are ground breaking, i.e. that haven't been done by that organization before
- and so on.

Whatever you choose to do, the steps are laid out here again.
Check:

- Does the amount of effort

associated with the particular job seem reasonable?

- Do the calendar days on which the job is due to be carried out make sense? For instance, have tasks been scheduled on public holidays or are unrealistic assumptions being made about what will get done over a holiday season like Christmas?
- Does each job have somebody's name against it

not the name of an organization but the person who will cause the work to be done?

The same comments made in the previous section about paralleling of tasks also apply here. Are tasks which actually have a dependency between them being run in parallel?

All of these tests applied to each job will generate review comments of the plan.

Finally, if you wanted to, you could do a simple risk analysis

using the four parameters from the First Law of Project Management:

- Functionality what is to be done
- Effort how much work is involved
- Delivery date (or elapsed time) when the job must be done
- Quality how well it is done

You would do this by asking questions like:

- What happens if the job runs over? (Elapsed time)
- Can the job be shortened? (Elapsed time)
- What happens if the job is bigger than we estimated? (Effort)
- What would be the effect of adding more people? (Effort)

- Is this job doing something which is crucial to the project? If not, could it be put into a later delivery? (Functionality)
 - What happens if this job isn't done perfectly? (Quality)
-
-

CASE STUDY 10

Scenario

You receive a project plan to review. It is a document of about 15 pages in length. The pages are unnumbered and there is no table of contents. Looking through it you find it has three sections with the following titles:

- Introduction
- Technical requirements
- A hand-drawn Gantt chart

The section entitled technical requirements is quite detailed. As you read it, you come to understand that the plan is to implement a system that will interface to some of the organization's other systems.

The system will be bought in, having been tailored by the supplier. Some development work will also be required on the organization's side. The new system will replace an existing system. The plan shows the project beginning on December 12, 2000 and being declared live by June 28, 2001.

The section called Technical requirements describes:

- the system that the project will deliver to the users
- the software, documentation, training and support services that form part of the project
- a period of parallel running and conditions that must be satisfied before the system can be declared live

- the hardware and software environment in which the system will be developed and run

This section also lays heavy emphasis on the users being involved at all stages of the system specification, design and development.

The Gantt chart is partially shown in [Figure 14.7](#). The horizontal bars indicate detailed breakdowns of each of the high-level tasks. The breakdowns are very detailed for December and January, and less so thereafter, but the Gantt chart gives the impression of being rich in detail. When you assess it using the WBS analysis, little or nothing seems to have been forgotten. Each task in the WBS has initials against it, showing who will carry out the task. Each task also has amounts of effort shown against it. Notes to the Gantt chart explain how these estimates were arrived at and

also explain the basis that each individual is involved in the project full-time, two days per week, required only for reviews, etc.

Figure 14.7.

	2000 Dec	2001 Jan	Feb	Mar	Apr	May	Jun	Jul
--	-------------	-------------	-----	-----	-----	-----	-----	-----

START

12/12
*

1. Write RFP

- write doc. AB, CD
- review doc. EF, GH, IJ
- sign off EF, GH, IJ
- send to suppliers AB

2. Choose supplier

-
-
-

3. Supplier development

-
-

4. Dublin Cor. development

-
-

5. Integration

-
-

6. Testing

-
-

7. Documentation

-
-

8. Installation

-
-

9. Training

10. Parallel run

11. Live running

THE END

*28/6

What would be your assessment of this plan?

Analysis

First level checks

Contents analysis

The plan contains three sections, whereas our proposed ideal table of contents contained thirteen! On the face of it, this doesn't seem like a very promising start, but we decide to persevere just a little to see what else we can find out.

- 1. *Statement of what is to be done.***
This turns out to be covered adequately in the technical requirements.

- *Deliverables.* Covered in the technical requirements.
- *Completion criteria.* Covered in the technical requirements.
- *Acceptance criteria.* Covered in the technical requirements.
- *Work breakdown structure.* The tasks down the left-hand side of the Gantt chart.
- *Gantt chart.* Present.
- *Milestones.* Can be deduced from the Gantt chart.
- *Resources required.* Human resources are present in the notes to the Gantt chart. Other resources (computers, software, etc.) are covered in the

technical requirements.

- *Resource loading.* Not stated explicitly but could be deduced from the previous section.
- *Project budget.* Not stated.
- *Project organization chart.* Not stated. The implication from the plan is that the author is the project manager.
- *Backup plan/margin for error.* If, on June 28, the system isn't ready, then the existing manual system can carry on as before. It may not be a very elegant backup plan, but it'll work!

WBS analysis

See point 5 above. Looks OK.

Gantt chart analysis: high-level

overview

Looking at the Gantt chart we see exactly the kind of high-level overview we are looking for. We can see:

- the main phases of the project and exactly when each phase starts and ends
- which phases depend on one another and which can proceed in parallel
- the amount of work in each phase

Resource analysis

A quick look at the Gantt chart shows that the project is due to start the week before Christmas, and a high level of activity is assumed between then and the end of the year. Cultures vary, as do organizations. However, in Europe, it would be unlikely

for such a schedule to hold true, and so this part of the schedule would probably have to be amended. This might have serious implications for the June 28 deadline. Analyzing the critical path on the Gantt chart would determine (a) whether or not this was the case, and (b) what could be done about it. As for Check 2, all jobs have people's names against them.

PSI analysis

Finally, let us calculate the PSI of the project.

- 1. Goal clearly defined. Yes, but the detail needs to be filled out. This will be done during the RFP and ultimately the development process. Score, say, 6 out of 20. Remember, a low score can either be a reflection of (a) a trouble spot, or (b) a result of where you are in the project's***

life. In this case, it is the latter.

- *List of jobs.* Yes, but the list is necessarily incomplete given where we are in the project. Say 6 out of 20 again.
- *One leader.* Yes, assuming the author of the plan acts as the kind of leader we require, i.e. the trail boss. Score 10, maximum points.
- *People assigned to jobs.* Yes, though because the job list is incomplete (see 2 above), this is as well. Score 3 out of 10 at this stage in the project.
- *Backup plan/margin for error.* Yes, albeit an inelegant one. Score 7, say, out of 10.

Total from the planning phase is 32. As we already noted 40 is the threshold for a successful project, so we're not there yet.

However, once the project has proceeded through the RFP stage we should be in good shape to go forward.

So, based on our initial quick assessment, despite an unpromising appearance, and some omissions, this plan appears to be in good shape. The main problem in it at the moment is the one with Christmas.

We could now go on to do:

- schedule and effort analysis
- critical path analysis of the Gantt chart
- resource loading analysis (Checks 3 and 4)

and among other things, this would tackle the Christmas problem.

Finally, if we chose to, we could do a Gantt chart analysis of the entire project.

CASE STUDY 11

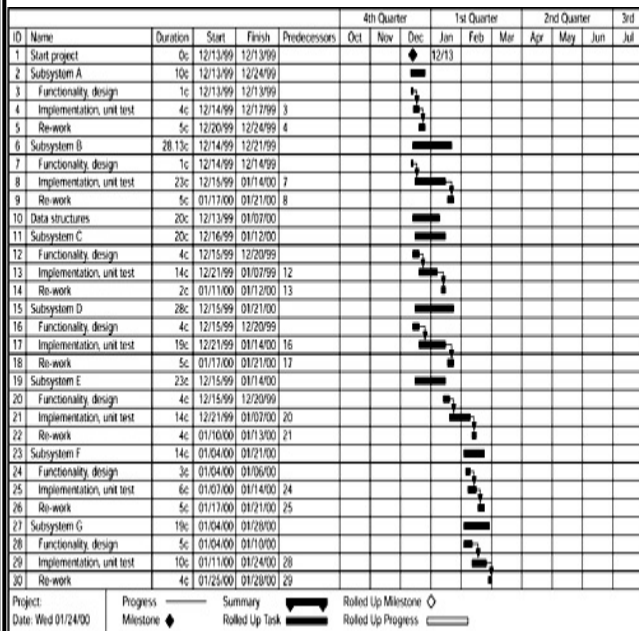
Scenario

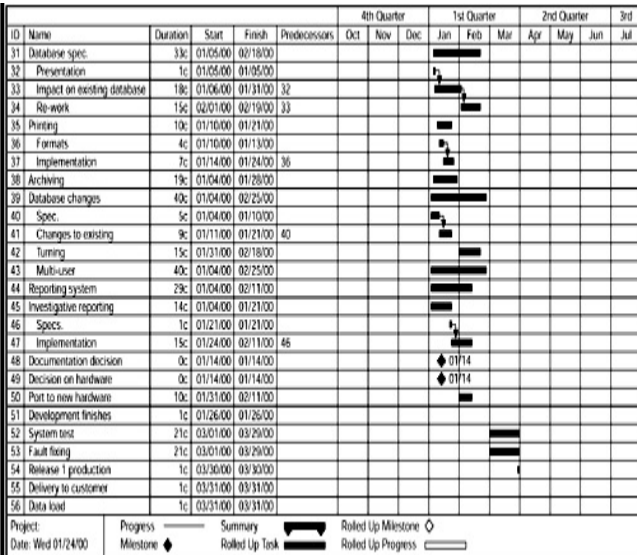
You receive a project plan to review. It is about 30 pages long. The plan is for a project to develop a piece of software, deliver it to a user site, install it and load user data on it by March 31, 2000.

Checking the contents you find that it has sections corresponding to *all* of the items on our table of contents checklist. Each of these sections looks fairly substantial and contains the kinds of things we talked about. In particular, a leader for the project is named.

The Gantt chart, which contains the WBS under the heading "Name," for the project is shown in [Figure 14.8](#).

Figure 14.8.





What is your assessment of this plan?

Analysis

First level checks

Contents analysis

As described in the scenario, it appears to be OK.

WBS analysis

The following appear to be missing from this WBS:

- any kind of requirements analysis or definition
- any kind of high-level design
- software integration
- development of tests (as opposed to carrying out the testing)
- project management

- configuration management and other project support activities
- production of documentation
- anything significant on installation or training

Gantt chart analysis: high-level overview

Non-existent.

Resource analysis

The problem with the Christmas holiday season also seems to be hitting this project (Check 1). All the jobs do have people's names against them (Check 2).

PSI analysis

1. Goal clearly defined. The goal of

the project really cannot be defined clearly if there are no activities (requirements analysis or high-level design) to define it! Assuming there is some requirements definition in the text of the document we will give them 10 out of 20.

- *List of jobs.* The list of jobs, the WBS, is to put it charitably not very good. Say 10 out of 20 again.
- *One leader.* The plan says that there is, so let's accept that. Score 10 out of 20.
- *People assigned to jobs.* Yes, they have been. But because the job list is a mess, this assignation will be flawed too. Score half marks, as we did for point 2 above, 5 out of 10.

- *Backup plan/margin for error.* Doesn't look good. The plan shows the system delivered and data load (job 56) taking place on March 31, 2000. If everything doesn't happen exactly on schedule, then the March 31 date will pass with nothing happening at the user site. Score 15 (i.e subtract fifteen), maximum score here is 10.

Total here is 20. With 40 as the threshold and with no jobs in the plan which might increase the goal and job list scores, this project would look as though it has very depressing prospects indeed.

CASE STUDY 12

Scenario

You receive a project plan to review. It is about 30 pages long. The plan is for a project to develop and release a system. The document makes it clear that the plan is a very aggressive one, that the project will start on May 1 and must finish at August 31 at the latest. The plan is based on a requirements document which has already been signed off.

Checking the contents you find that it has sections corresponding to *all* of the items on our table of contents checklist. Each of these sections looks fairly substantial and contains the kinds of things we talked about. A leader for the project is named.

Checking the WBS you find that there are lots of omissions and inaccuracies and you note these down.

You look for a high-level overview Gantt chart and find it (reproduced in [Figure 14.9](#)). It is hand drawn but nonetheless appears to have the major blocks in it. A more detailed Gantt chart (not reproduced here, but containing all the kinds of things we expect) accompanies the high-level overview.

Figure 14.9.

May

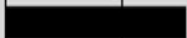
Jun

Jul

Aug

Sep

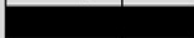
Design



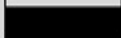
Code unit test



Integration



System test



A schedule and effort analysis, resource analysis histogram showing everyone scheduled at or less than 100 percent and assumptions on people's availability are all included among the pages of the plan. There is also a note in the plan to the effect that all of the project team members' summer holidays have been accounted for in the estimates and schedules.

What is your assessment of this plan?

Analysis

First level checks

Contents analysis

As described in the scenario, it appears to be OK.

WBS analysis

The results of the WBS analysis are already noted in the scenario.

Gantt chart analysis: high-level overview

There is certainly a high-level overview with a more detailed Gantt chart backing it up. However, there is a major flaw in the high-level overview: the four activities in it overlap one another to a highly suspect degree.

- Software is being written while the system is being designed. This is always a dangerous thing to do.
- Integration of the software begins before all of the software has been written. While this could indicate good forward planning on the part of

the project manager, it can sometimes be an indication that somebody is merely trying to squeeze everything into the allotted time. This would merit further investigation at a lower level of detail at the critical path level.

- The system test isn't entirely what it seems. A system test should do exactly that: it should test the entire system. Yet, in this plan, system test begins before the system is fully available. Again this might imply great foresight on the part of the project manager. Unfortunately it more often implies the presence of Parkinson's Law "Work expands (or contracts!) to fill the time allotted."

Resource analysis

As we noted in the scenario, the summer holiday season has been allowed for (Check 1). Also the who's doing what is contained in the more detailed Gantt chart (Check 2).

PSI analysis

1. *Goal clearly defined.* By paralleling the design and writing of the software, there is a fair chance that the goal of the project will never be clear; that what the software people write will not be what the designers wanted. Give it, say, 12 out of 20.

- *List of jobs.* The list of jobs, the WBS, is, as stated in the scenario, not very good. Say 12 out of 20 again.
- *One leader.* The plan says that there is,

so let's accept that. Score 10 full points.

- *People assigned to jobs.* If the job list is flawed, this will be too. Score in the same proportion to point 2 (above), 6 out of 10.
- *Backup plan/margin for error.* Doesn't look good. The plan shows every sign of Parkinsonian planning. Not only is there probably no margin for error, but the plan as it stands is probably not based on reality to begin with. Score 15 (i.e. subtract fifteen), maximum score is 10.

Total here is 25. With 40 as the threshold and with no jobs in the plan which might increase the goal and job list scores, this project would look as though it has very depressing prospects indeed.

As we have seen in all three of these case studies, you can often but not always

find out all you need to know about the plan by applying the five small checks that we talked about earlier, i.e. the ones that take 30 or so minutes. Just to remind you, these are:

CHECK	EFFORT INVOLVED
Contents	Small
WBS	Small
Gantt overview	Small
Resource analysis checks	Small
PSI	Small

Second level checks

For Case Study 12, just for interest, we will look at what we can find out by applying one of the tests at the second level. Thus, out of the three tests at this level, i.e. we will apply schedule and effort analysis.

CHECK	EFFORT INVOLVED
Schedule and effort	Medium
Gantt critical path	Medium
Resource loading 3 and 4	Medium

Schedule and effort analysis

The schedule and effort analysis, given in

the plan, is shown in [Figure 14.10](#). Adding our comparison figures is shown in [Figure 14.11](#). An interpretation of the differences between these two sets of figures is as follows.

Figure 14.10.

Estimating score card

Project

Case study 3

Author

Revision No.

Date

Phase	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%
Design		25		12								
Code & test (3a)		41		75								
Integration		17		3								
System test		17		10								

Figure 14.11.

Estimating score card

Project

Case study 3

Author

Revision No.

Date

Phase	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%	Schedule (months)	%	Effort (MM)	%
Design		25		12	A	26		18				
Code & test (3a)		41		75	B	36		36				
Integration		17		3	C	18		9				
System test		17		10	D	18		20				

1. The amount of effort going into design would appear to be too low. On top of the paralleling going on between Design and Code and unit test, this further evidence is very disturbing.

- This figure of 75 percent in Code and unit test is bordering on the insane!
- So is the 3 percent in Integration.
- As is the 10 percent in System test.

This check only adds weight, if it were needed, to the evidence we already have.

GUIDELINES FOR WRITING PROJECT PLANS

If you are charged with writing a Project Plan then the following should help to ensure that you make the best possible job of it. (This is presented as an alternative to the one in the Introduction to [Part 2](#).) Your plan should contain the following elements:

1. Table of contents.

- Management summary.
- Statement of what is to be done. (The goal of the project.)
- Deliverables. (Also part of the goal.) These may come under a number of headings:
 - Software
 - Hardware
 - Documentation
 - Services

- Completion criteria. How do we know when the project is over? (A key part of the goal.)
- Acceptance criteria. How does our customer decide the project is over? (Also a key part of the goal.)

The next nine items are all part of the plan and the backup plan.

7. Work breakdown structure. A list of all the jobs that have to be done to complete the

project together with estimates of their associated effort and any assumptions made.

- A Gantt chart. Showing the WBS phased over time, and stating any assumptions made. The Gantt chart should also show clearly the critical path of the project.
- A breakdown of the effort and schedule in the project by project phase.
- Milestones. A list of key

dates extracted from the Gantt chart.

- Resources required. Human and otherwise.
- Resource loading. How the resources are phased over time and how the availability of people's time has been calculated.
- Project budget (optional). Derived from the effort in the WBS and the section on resources.

- Project organization chart. Among other things, this serves to identify the project leader.
- Backup plan/margin for error. Sometimes called risk analysis. The project plan represents a prediction of how things will go in the future with regard to this project. The intention of this section is to show that the author of the plan has given some thought to what she will do when what happens in reality differs from the project plan.

Part 5: THE REST OF THE WHEREWITHAL

Chapter 15.

RESOLVING

ISSUES: PROBLEM

SOLVING AND

DECISION MAKING

[INTRODUCTION](#)

[PROBLEM-SOLVING](#)
[METHOD](#)

INTRODUCTION

In [Parts 1](#) and [2](#) of this book I have presented a general method for leading a project. We have talked about how to plan it, what things to do during the execution of the plan, and how to bring the project to a successful conclusion. The method presented was conceptually very simple and pretty much common sense.

However, life is never that

simple. Countless issues, large and small, will arise during your project. Not a day will go by when there won't be problems requiring solutions; every day, you will make decisions which will affect the outcome of the project.

This part of the book presents a general method and a toolbox of techniques that you can use to help resolve these issues, to solve problems and to make decisions. Armed with your commonsense method in one hand and your problem- solving method in the other, you should

be the equal of any issue which arises to confront you.

In this chapter I will list these problem-solving techniques and, where useful, give examples of their application. There are no right and wrong situations when one or other technique should be applied. As we have seen, projects are complex organisms, and any issue is likely to be complicated, with numerous factors to be balanced. Several of the techniques which follow may all resolve the issue. Only you can decide which resolution

is right for your project. If this chapter had a subtitle it would be "There's more than one way to skin a cat."

PROBLEM-SOLVING METHOD

The general problem-solving method we will use contains the following four steps:

1. What's the problem?

- What's the ideal solution?
- Identify a range of solutions.
- Implement one or more of the solutions identified.

We will now look at ways to carry out each of these steps.

What's the problem?

The first thing you have to do is to know what the issue is. Sometimes you can set off trying to resolve an issue, before you have really understood it. Therefore you need to be aware that the following can happen:

- 1. sometimes the stated issue is not the real**

issue

- sometimes a problem is stated by stating its solution
- sometimes, even if an issue is stated correctly, you need to pull back and look at the bigger picture in order to arrive at a resolution

Often, a good way to identify an issue is to state it. Simply stating the issue committing it to paper, or talking through it aloud can both crystallize the issue and, sometimes,

throw up pointers as to how to move forward.

What's the ideal solution?

You may never be able to achieve it, but it's an invaluable thing to know. What would be the best outcome? It's related to our old hoary question, what do you want to do? What's the ideal solution? If there were no restrictions or constraints, what would you do? You'd be surprised how many times you can come out

of the issue with a resolution which is not far off the ideal one.

Identify a range of solutions

The following are all designed to assist you in identifying possible solutions to your problem or issue.

Chess

We are told that world-class chess players can see a game many moves ahead. They know that if they make a certain move, their opponent will probably do A, then they will do B, their opponent will do C, and so on. Often you can resolve something just by exploring your choices, thinking things through and looking at what is likely to happen.

If you do go one way, then here are the alternatives, and depending on what happens here, further alternatives present themselves. I regularly

draw charts on the board showing the decision to be made, the options emanating from that decision, and the subsequent scenarios that become possible. These can end up looking like a large tree in winter, there are so many branches, but it gives you a very clear picture of what's facing you. Two other points to note are:

- If one particular path is more advantageous to you, you can see it very quickly and start to move things

down that path by making the other paths seems less attractive to other people.

- You can save yourself a lot of time because moves may become redundant, i.e. irrespective of what happens as a result of the next decision(s), you will still end up in the same place further down the line.

The brownie points game

Brownie points represent your popularity ratings with other people. Each person connected with the project will have given you a brownie point rating, either consciously or subconsciously. At any given time, some ratings will be high and some not so. The brownie points game says that you will choose to resolve an issue by doing it in such a way as to maximize your brownie points rating with a person or persons.

Shortly before the Gulf War started, Saddam Hussein gave

all his soldiers salary increases. We don't know if he had a morale problem, but if that was the problem he was addressing, he chose to address it by playing the brownie points game.

People who suck up to their bosses are constantly playing the brownie points game with them. I once had a boss who told me, at my appraisal, that he was giving me a 10 percent rise. None of my peers were getting such an enormous rise, he told me, only me, and mine was for a performance above

and beyond the call of duty. As it turned out, at least two of my peers also got 10 percent rises. My boss was playing the brownie points game.

It is a feature of the brownie points game that it can often achieve spectacularly good results. It is also a feature that those results can be very short term and have negative effects in the long run.

Brainstorming

Brainstorming is a wonderful technique. The idea is to get a bunch of people together and then write down all the possible ways of resolving an issue.

There are two phases. In the first phase all ideas, no matter how far-fetched, are acceptable. Nothing is ruled out. Any idea which appears to resolve the issue is added to the list. Note that you can use all the techniques in this section to come up with ideas. In the second phase, you run back down the list and try to extract the best idea or set of

ideas.

Do nothing

When we think of projects we think of frenzied levels of activity while we work to get the project done. What if an issue could be resolved by doing nothing? Got its attractions? You bet it has. Strange as it may seem, I have identified five variants of the do nothing approach. They are:

- 1. Literally do nothing.***

Whether you genuinely believe the issue will go away, or whether you just wish it would, isn't the point; you decide to take no action.

- *Take another reading.* You resolve to do nothing, and look at the issue again some short time in the future. Depending on this next reading, you may decide to take action or if things appear to be going the way you wanted, to wait and read again.

- *Chaff.* Chaff is strips of aluminum dropped by war planes to confuse enemy radar into thinking there are lots of your side's planes. In our context it implies that you give the impression of doing something about an issue when in fact you're doing nothing. For example, you may have inside information that an issue which is causing your team major grief is about to be resolved. You don't want them to see you doing nothing because it's such a hot thing with them, but at the same

time you don't want to waste any time on it because you know resolution is in the pipeline. You therefore send forth signals implying a major level of activity on the issue, when in fact you're doing absolutely nothing. This is chaff.

- *Recce* (reconnaissance). You need to get more information. While you are gathering this information, you do nothing about the issue itself.
- *Stonewalling*. This is where

you decide to do nothing and want it known that you're doing nothing. The classic has to be when someone asks you for a rise, and you have decided they're not getting one. Stonewalling basically involves saying no until they go away!

Truth tables

Truth tables are a technique taken from computer science and are very useful in analyzing a problem. Here is a

somewhat contrived example. I would like to give up my job and become an organic farmer. Is there a way I can do this without taking a hefty drop in my standard of living? The issue is composed of two factors here (becoming an organic farmer and not taking a drop in income) but any number of factors can be considered; the truth table just gets bigger. Each factor has two possibilities, either it happens (yes, Y) or it doesn't (no, N). The truth table is then drawn as follows:

Option:	1	2	3	4
Become an organic farmer	Y	Y	N	N
Take a drop in income	Y	N	Y	N

I then have four choices.

- Option 1 means that I become an organic farmer, but I take a drop in income. Doesn't solve the problem.
- Option 2 means that I become an organic farmer and take no drop in income: the ideal solution.

- Option 3 is not to become an organic farmer, but to take a drop in income. This is clearly the worst scenario, as it takes me further away from my defined goal, to be an organic farmer and suffer no loss in income.
- Option 4 is where I am at the moment: I'm not an organic farmer but my income is reasonable.

This analysis tells me that in terms of my defined goal, I

have not reached it, but neither am I as far away from it as I could be. As I said, it's a somewhat contrived example, but still the insight is interesting.

The Ronald Reagan method

Sometimes, in your projects, an issue will come along and you don't really care how it is resolved; all that is important to you is that it *is* resolved,

that some decision gets made. For example, it might be that until a decision is made, your project is held up. Some years ago there were stories circulating that Ronald Reagan made all his decisions, large or small, in the following way.

One sheet of paper was handed to him; an issue was described briefly, in simple sentences with short words. The decision to be made was stated at the bottom; and two boxes, one marked "YES" and one marked "NO," were drawn. A note said "Please tick one." The word

"one" was underlined.

I have often used this box method at meetings. It is useful in cases where the meeting is snarled up on an issue and you want to push the meeting forward. State the decision to be made, and get people to choose.

Gradually warm up an issue

This is a very effective

technique and one which really proves the truth of the saying that there's more than one way to skin a cat. Most of the methods so far, have one thing in common: they attempt to resolve the issue there and then. Gradually warming it up resolves it over a longer timeframe.

You have an issue, you know it's going to be a thorny one, and you're fairly sure that if you went at it head-on, you'd be shot down. The first thing then is just to get it on the table. You say perhaps, "I don't

want to talk about this today, but I'd like to deal with it some time in the future." This gets the other side thinking about it. You then take it a stage further: maybe you have an informal discussion about it over lunch or coffee. Following this perhaps you write a proposal and see how that flies. This process is continued until you get what you want. If, at any stage, you find it's not going the way you want it to go, you back off, wait and then move forward again gradually.

I have used this method over

short (1 – 2 weeks) and long (1 year) periods. It is highly effective.

"What would you do in my position?"

If nothing else, it provides you with another option for resolving the issue. Note that it is another way of asking what the other person's ideal solution is.

Put yourself in the other person's position

See things from the other person's point of view. Any good salesman will tell you that if you can do this — see your customer's point of view, his problem, his concerns, his fears, his objective, his motivation — then you're well on the way to closing the sale.

Treat it as a project

Apply structured project management!

Do a deal

"I'll do this if you do that."

"Here's what I'll do for you;
now, what'll you do for me?"

Losing your temper

Can often be disastrous, but
can sometimes be highly
effective, particularly if, like

me, you're a person who doesn't often lose it. Several times, I have had to pretend I was losing my temper, and have wondered whether I should be considering a career as an actor!

Pep talk

Sometimes all the person needs is a bit of a lift. A pep talk is what the coach gives the team at half-time. It is usually very obvious if someone is

doing it to you, but depending on your state of mind, it may be enough. It is a very simplistic approach and you need to be sure that it has worked, because if not, you have not resolved the issue.

Selling something unpleasant

For the particular case where you have to sell something unpleasant to somebody a nasty or boring assignment, for

example you can use a technique from selling. This is the idea that you don't offer your customer a choice whether or not to buy; what you offer him is a choice of ways to buy. Thus you might offer a person a number of different ways he can carry out the particular assignment.

Grovel

This may be a hard one to swallow, but there are times,

particularly when you've cocked something up, that an apology is the only way. Many people have problems apologizing to subordinates, and apologies aren't easy at the best of times. Sometimes, unfortunately, it's the best way.

I have saved two really fancy ways of identifying possible solutions until the end. They are very complicated, risky in that they can backfire, but hugely enjoyable if you can pull them off. You might use them just for the hell of it, to test your skills; or to show off your

power to other people peers, staff, the opposition. Don't use them too often, save them for a special treat for yourself.

Feigned retreat

Feigned retreat, as you might imagine, is a military term. The classic example of its use is the Battle of Hastings in 1066 ([Macdonald, 1984](#)). It is an example of what [Clausewitz \(1984\)](#) calls a stratagem. What this means is that by your

actions you mislead the other side into thinking you're going to do something different from what you actually plan to do.

Feigned retreat is beautiful when it works, as in the following example. (Some of the details have been changed to ensure anonymity, but you'll get the gist.)

I once worked for a company, and my division bid for a large project from another company. We wanted that project really badly. The other company messed around for a long time

they were going to do it themselves, they were going to subcontract it elsewhere. Our proposal was totally unacceptable: they couldn't believe it was going to take as long as we said, or that it would need so many people.

There was to be one last crunch meeting. It began with the same old stuff they had to have the product sooner, they could easily do it themselves, did we have anything new to offer? Yes, we said, you guys take it, we don't want it. We can help if you like, perhaps

sell you the odd piece of consultancy. And we've done a lot of work in estimating this, so we could give you some help there, because we think some of your estimates may be a little bit out (they were wildly out!). So why don't you guys go away, and come back to us if we can help. This was all said in the most civil of terms. Inside, we were squirming; if we lost the job, we had major problems. However, we didn't. After a hurried conference, the other guys came back and awarded us the contract.

Here's another example of what can happen when you try to execute a feigned retreat. In military situations, feigned retreats are considered very difficult maneuvers to carry out. This is because a feigned retreat can, all too quickly, become a real one.

A guy I knew — let's call him Albert — was the king-pin in an organization for a long time. His boss was ineffectual, and Albert did pretty much as he wanted. Then a new boss arrived, and things started to change; a lot of Albert's power

started to be eroded by his new boss. Albert was totally cheesed off, and chose the following course of action. He knew he was crucial to the running of the place, and in this he was quite correct: he was the most skilled technical person there, and extremely difficult to replace. He would get another job offer, and then attempt a feigned retreat on his boss let's say he's Bert.

Having secured the job offer, Albert went in for the meeting. "I really like working here, Bert, and I like working with

you, but I'm unhappy with some things and I'd like to discuss what we can do about them." Albert went through the various things responsibilities, position in relation to his peers, grading, salary. Bert pretty much stonewalled, a few minor concessions, but nothing of any substance.

Albert repeated his position this time somewhat more forcefully how he really liked so many things about the place, but unless his perfectly reasonable demands could be

sorted out, he would be forced to leave, and he didn't want to. Bert continued to stonewall. OK, Albert said, he hadn't wanted to do this, but Bert was leaving him no option.

Albert explained that he had another job much more responsibility, better salary, more prospects and that he would be forced to take it unless he and Bert could settle their differences. Bert rose, extended a hand across the table, and said: "Why, Albert, congratulations. You know we'll be sorry to lose you, but it

sounds like a wonderful opportunity that will really advance your career." Albert left. He had never wanted to, and ended up having to relocate his family 3,000 miles.

Double envelopment (the pincer movement)

Double envelopment involves coming round an opponent from both sides so as to surround him; the phrase

pincer movement describes it graphically. Again, it is a military term, and the classic example is the Battle of Cannae in 216 BC ([Macdonald, 1984](#)) when Hannibal defeated a Roman army in Italy. A more recent example from our own times was the series of maneuvers carried out by the Allies which brought the Gulf War to a conclusion.

In a pincer movement as we're applying it, you decide to resolve the issue by identifying a number of solutions, and then apply two or more of them

simultaneously. The thinking is that at least one of your attacks should work, and if they all work it provides a dazzling display of your powers. The downside is that if none of them works, you end up a little naked and unarmed, that is, you've used up all your ammunition.

You can contrive a pincer movement by taking your plan and fallback plan, and applying them simultaneously. The trouble with this is that you then have no fallback plan. You could live your whole life and

never do a pincer movement and still be highly successful in resolving issues. I mention pincer movements here mainly because they're great fun, and a good test of your skills.

Implement one or more solutions

Here are three methods to help you to pick a solution from the group of solutions you have generated.

Write down the pros and cons

Take a piece of paper, write down the problem, write down the action you intend to take. Then draw a vertical line down the center of the page and note the advantages on one side and the disadvantages on the other.

Betting

It is not the intention of this

book to turn anyone into a compulsive gambler, but betting is a technique I have found very useful when dealing with issues. You can bet on anything the chances of a milestone being met, the project completing successfully, a key person leaving the team you name it and you can bet on it. Generally it is best if the bets are small (less than \$5) and short term they can be resolved within a matter of weeks. Having to bet does three things:

- it causes both parties to think very seriously about any situation, and the odds each attaches to a particular thing happening
- if the thing is within our control it causes us to make a major effort to ensure that it goes the way we wish it to go
- it establishes a level of confidence about the issue

Give yourself a

present

Promise yourself a little present if the solution(s) you selected works out as you intended.

Finally, here are a few other rules of thumb that you should also bear in mind when resolving issues.

**Don't dump on them
at least until they
dump on you**

If one of the methods you have identified to resolve an issue involves taking a dump on another person, don't choose that method unless they have first dumped on you.

(Statesmen of the world take note.) Thus the issue remains at a civil level unless the other side opens hostilities.

Choose the right moment: timing is everything

One of my customers can be quite grumpy to be back in work after the weekend. I never call him on Monday mornings!

Get the monkey off your back

However you choose to resolve an issue, it is nice if you can end up with a resolution where the issue is no longer your problem. This is what is called getting the monkey off your

back. All other things being equal, a method which moves the monkey off you is the one to go for.

**If one technique
doesn't work try
another one**

Any issue can be resolved. If you find one technique isn't cutting it for you, try a different one.

If it's potentially harmful sleep on it

Maybe you're going to fire someone, or take a giant dump on somebody, or do something for which you might get into trouble. Whatever it is, if it has serious negative consequences, mull it over during the drive or train ride home. Sleep on it. Then, in the morning, if you still think it is the best thing to do, do it.

Don't let things fester

Don't let things fester. By all means wait and choose the right moment, but don't procrastinate and let a thing run on. The chances are it will only get worse. Resolve the issue as quickly as is practicable.

Chapter 16. COPING WITH STRESS

INTRODUCTION

WAYS TO REDUCE STRESS

INTRODUCTION

Stress occurs when you're not in control, when outside events are driving you instead of you driving events.

Much of this book so far has been about trying to get to a situation where you are in control; where the uncertainty and guesswork surrounding your project have been reduced to a minimum, and where you know you have done all that can be done at any particular

point. In [Chapter 15](#) we looked at ways of overcoming problems and difficulties that might be causing you stress. All of the techniques presented in that chapter should cause a significant reduction in your stress levels.

However, even the bravest of us are still prone to anxiety, worry, stress, and so this chapter talks about some other things you can do. In general these things won't remove the cause of the stress — it is up to other parts of this book to do that for you. What the methods

in this chapter will do will be to change your attitude to the issue in question, and hopefully reduce your anxiety about it. Here's a quote to set the tone of this chapter.

You can think about your problems or you can worry about them, and there is a vast difference between the two. Worry is thinking that has turned toxic. It is jarring music that goes round and round and never comes to either a climax or conclusion.

Thinking works its way through problems to conclusions and decisions; worry leaves you in a state of tensely suspended animation. When you worry you go over the same ground endlessly and come out the same place you started.

Thinking makes you progress from one place to another; worry remains static. The problem of life is to change worry into thinking and anxiety into a creative action.

Harold B. Walker

With this idea in our back pocket, this chapter looks at twelve ways of reducing and controlling stress.

WAYS TO REDUCE STRESS

Who has the ball?

If you are stressed on a project, then you need to locate the thing that is causing the stress. Having located it, there are only two possibilities: either you can do something about it or you can't. If you can, then you have the ball and you run with it. Note that this may involve starting another

project.

If you can't do anything about it — and by this I mean you have tried all the techniques in [Chapter 15](#) on problem solving — if you really can't do anything about it, then somebody else has the ball. Having identified who this is, and note that it may be Fate (or God, whichever you prefer), and if you still can't see anything you can do to influence the outcome, then *there is nothing you can do*. And by nothing I mean nothing. In particular, don't worry about

it. Identify the event(s) that will cause the ball to come back to you. Wait until it comes, as soon or later it will, and then off you go, back into action.

Keep a sense of proportion, or, there's always someone worse off than you

Before I finish this chapter thousands of innocent people will have died throughout the world. They will die of hunger,

disease, torture, execution, neglect, abuse, loneliness. It's important to keep a sense of proportion. In general, unless you are involved in certain types of projects, for example military, medical, you almost certainly won't be doing life-threatening things.

Many of the things that face you in your projects don't add up to a hill of beans (as Humphrey Bogart said) in the context of some of the real problems in the world. The next time you're feeling stressed pick up the paper and read the

world news. Alternatively, you could turn (as promised) to Winnie the Pooh. This is our friend Eeyore again, in conversation with Christopher Robin.

'Hallo, Eeyore,' said Christopher Robin, as he opened the door and came out.

'How are you? 'It's snowing still,' said Eeyore gloomily.

'So it is.'

'And freezing.'

'Is it?'

'Yes,' said Eeyore.

'However,' he said,
brightening up a little, 'we
haven't had an
earthquake lately.'

Try treating it as a game

If you are stressed it means
you feel that something you
hold dear is under threat. It
could be your life, your job,
your marriage, your career,

your reputation, your home, your family. One way to behave is to try treating it as a game. Suppose the stakes weren't so high. Suppose it were just a game and in a couple of hours' time, the game will be over and reality will return. Now, with this different viewpoint, do you get any new insights into how to move forward?

Exercise

I didn't think this one up.
Doctors recommend exercise,

particularly aerobic ones swimming, cycling, running, rowing, and so on the very rhythmic exercises, as a cure for stress. Yoga and any kind of meditation are also very good.

See it a year from now

Take the thing that is causing you stress and imagine how it will look a week, a month, a year from now. Will it really seem that important? Do you think you will actually be able to remember it? Pretend it is

some time in the future and see how the picture looks.

Has it bottomed out?

In calculus there's this idea of taking a point on a curve and comparing it against a closely neighboring point. From this you can determine whether the curve is ascending or descending. (I'm being fairly loose in explaining the mathematics of all of this!) You can use this idea by taking a reading, as it were, each day,

and determining whether your situation is improving or not.

Compare today to yesterday. If things seem even infinitesimally better then maybe the situation has reached rock bottom and is now starting to improve. This one is particularly good for debts! and, if the curve is going the wrong way, try another method.

Get gates to your house

My house has gates and a short drive leading up to it. Often when I come home in the evening and close the gates, I have the feeling of locking the problems of the world out. Tomorrow, when I go through the gates, the problems will still be there, but tonight, I'm alive, I've got my family. I have a little oasis in the desert. Tomorrow is still there, but it is in the future, somewhere out there. In the meantime I can enjoy what I've got.

The marathon runner

I used to run marathons. Because the thought of running 26 miles is so outrageous, there's this idea in marathon running that you forget about how far the distance is, and just focus on the next tree or house or telegraph pole. Sure, the rest of the race still remains to be run, but by taking a very short horizon, focusing on it and blocking out all the other, future stuff, you can enormously reduce the pressure on yourself.

Write down what the

problem is

A bit like stating the issue in [Chapter 18](#). Write the problem down, study it and see if that in itself is not therapy and gives you a fresh insight.

The world closes down at the weekend

Many of the institutions that cause us stress — banks, businesses, the government close down at the weekend. In

many countries, postal deliveries, which might bring further stress, stop sometime on Friday afternoon. This means you have a whole two and a bit days' respite before having to go back into the fray.

Talk to somebody

As the old saying goes, "A problem shared is a problem halved."

Keep a diary

Some novelist and can't remember who it was to save my life, or even find the reference (if someone can help, I'd appreciate it) said "I write to see what I think" (or words to that effect). Write, or more specifically keep a diary, to see what *you* think. Use it to analyze, to comment, to understand, to suggest ways forward. You may find it a help.

Chapter 17. PICKING THE RIGHT PEOPLE

INTRODUCTION

METHOD OF
INTERVIEWING

INTERVIEW QUESTIONS

INTRODUCTION

For many projects you will need to pick people and the question we discuss here is how to pick the right ones. This book talks about taking a project-oriented view of an endeavor. Some people do this instinctively. I believe that most others can be taught to do it without a great deal of difficulty. Indeed, that is what this book is trying to get the reader to do. In this chapter I will talk about interviewing, but you should

think of this in the sense of finding people for your project, as opposed to simply hiring people.

METHOD OF INTERVIEWING

In my last regular job, I was General Manager of a US multinational's European Software Development and Support center. There we adopted a two-stage approach to interviewing people. In the first stage I screened all the résumés and did preliminary interviews. (Given that we were over 60 people, and that we were very choosy about whom we picked, you can see

that this would have made for a hell of a lot of interviews in a year. The rationale was that I was the project leader and I needed to have a large say in who worked on my project.)

Companies all say that people are our most important resource, but how many of them see hiring and people-related issues as belonging to the personnel or human resources departments? People are how we get our projects done. That is why they are our most important resource.

The interviews were generally short (half an hour), and really only asked three questions:

- What have you done?
- What do you want to do?
- What are you like?

A result of the same person doing all the screening is that over a period of time, such a screening process guarantees a consistent type or types of person getting through to the second interviews. It may

result in people who would have been right, being excluded. On the other hand, if you've found a type or types that work for you, why change?

In the second interview, which could last 3 or 4 hours, the person got to meet as many of the other team members as possible: peers, subordinates, managers. We encouraged the person to walk around and talk to people, particularly to people doing a job similar to the one this person would be doing. The person did this in a totally uncensored way, that is, he

could talk to whoever he chose and the person to whom he talked could say whatever he liked. There was no rehearsal or *party line*. We took the view that we had nothing to hide, that people needed to see us warts and all. We wanted people to feel what it would be like to work in our team, and that was why the interview lasted so long and the candidate saw so many people: it was the closest they could get to us short of actually coming to work there.

INTERVIEW QUESTIONS

Here is how to go about getting answers to our three basic questions.

What have you done?

It is at this stage you find out what experience and qualifications the person has. For a very technical area there are certain technical skills that are essential. We look for people who describe their

career in terms of milestones or achievements, that is project-oriented people. Typical questions are:

- What was the greatest moment of your life?
- Tell me the three things you've enjoyed doing most in your life.
- What was your biggest setback?

Things to look for are:

- Punchy replies the person tells you the answer, gives you what you asked for and shuts up. Rambling replies make me very edgy, as they seem to me to be indicative of a jumbled or unstructured sort of mind.
- A feeling that the person has pushed through difficulties to get a job done.

What do you want to do?

The key question. All along we have talked of harnessing people's personal objectives to enable us to make our project happen. If the person doesn't know what they want, then I will almost always not hire them. If they don't know what they want, then how can you use them in your project — it may not be what they want?

Obviously, I won't just ask the bald question and if they can't answer kick them out. I'm prepared to probe, and have done so to an excruciating degree on occasions. However,

at the end of the day, you have to wrinkle it out of them.

Incredibly, this can sometimes be the first time they realized what they want to do. We ask questions like:

- What do you definitely not want?
- If you could pick any job in the world what would it be?
- How would you like your career to go?
- What drives you?

- Are you a task-oriented or people-oriented person?
- What do you do in your leisure time?
- What would someone else say about you a former boss, your girlfriend? It almost doesn't matter who it is it's just that if the person is hanging tough on this question, you need to get an answer.
- How hungry are you for the job?

You need to know what the person will be like when they show up on your team. You know what you want to do with your project, and what part you want him to play. The question is what does this fellow want to do?

What are you like?

Finally, you need to get a feel for the person's personality. Do you feel that they'll fit in with the general chemistry of your group?

- Describe your personality.
- Are you the kind of person that if I ask you to do a job I can regard it as done?
- What strengths and weaknesses would you be bringing to the table?
- Say something negative about yourself.
- What do you like to read?

Chapter 18.

NEGOTIATION

INTRODUCTION

PRINCIPLED NEGOTIATION

INTRODUCTION

Every one of us, in all aspects of our lives, has to negotiate. We do it over salary, when we buy things, in our personal lives, as part of our jobs; and we see it going on in world affairs where the stakes can sometimes be scarily high. We are familiar with what happens in negotiation: we take a position, the other side takes one and then both sides argue until a compromise is reached.

There is a wonderful little book called *Getting to Yes* ([Fisher and Ury, 1981](#)), based on work done at Harvard. This short book presents a method called *principled negotiation* or *negotiation on the merits*. The method has as its objective to produce a wise agreement. A wise agreement is defined as "one which meets the legitimate interests of each side to the extent possible, resolves conflicting interests fairly, is durable, and takes community interests into account."

PRINCIPLED NEGOTIATION

The principled negotiation method has four steps:

- 1. Separate the people from the problem.**

- Focus on interests, not positions.
- Generate a variety of possibilities before deciding what to do.

- Insist that the result be based on some objective standard.

Let us look at each of these in turn.

Separate the people from the problem

Often negotiations can end up as personal battles between the participants, where strong emotions and people's personalities obscure the real issues. Principled negotiation

tells us to disentangle the people from the issues; and to move from a confrontational situation to one in which the participants are working together to attack the problem as opposed to attacking each other. Some of the techniques from [Chapter 15](#) can be useful here, particularly the following:

- Exactly what is the issue?
- State the issue.
- Truth tables.

- What's the ideal solution (very useful)?
- "What would you do in my position?"
- Put yourself in the other person's position.
- Treat it as a project. Apply Step 1 of structured project management visualize what the goal is; set your eyes on the prize.

Focus on interests, not

positions

Most often we take up negotiating positions based on a number of factors (interests) which are important to us. In a conventional negotiation, the position can quickly come to obscure the interests.

Principled negotiation says to focus on the interests and not the positions. Again the techniques mentioned above can be used to help to do this.

Generate a variety of

possibilities before deciding what to do

In other words, brainstorm.

Insist that the result be based on some objective standard

In a conventional negotiation results are often arrived at by one side giving in, or the other side being intransigent. A different way is to agree in advance an objective standard

or set of criteria by which the outcome will be judged. Once this is done, nobody has to give in — both sides can use the standard as the definitive reference point. Examples of an objective standard are market value, precedent, equal treatment or expert opinion.

In this chapter I have tried to give you a feel for the method of principled negotiation. Like many of the other things in this book it smacks of just plain common sense. Instead of arguing the toss about something, we say "look, we

have this problem, how can we solve it to our mutual satisfaction?" It is a way that we know instinctively is:

- more effective
- more likely to be successful
- more likely to produce the best solution
- a damn sight more pleasant

than traditional negotiation.

Getting to Yes ([Fisher and Ury, 1981](#)) won't take you long to

read it's about half the length of this book. I urge you to get your hands on a copy.

Chapter 19.

MEETINGS

INTRODUCTION

THE MEETING ALARM

ORGANIZING AND
RUNNING MEETINGS

INTRODUCTION

A meeting can be an incredibly useful arena on a project. In a well-conducted meeting you can smash through obstacles, resolve issues and make decisions, and come out feeling like you really moved the project forward and that in paying today's salary, your employer really got value for his money. How many of us, though, attend meetings which:

- drift along

- appear to have no purpose
- have no agenda so you have no idea whether or not you're making progress if you didn't need to be there at all?
- appear as if you didn't need to be there at all?

This chapter contains a quick way to find out if the meeting you're being asked to attend will be such a meeting, and some hints about how meetings should be run so as to be of the

effective kind described in the opening paragraph.

THE MEETING ALARM

If somebody asks you to a meeting you should find out:

- What is the objective?
- What preparation is required?
- Why do you need to go?
(The first two questions should make this clear. If you have to ask this question, it is in itself a

danger sign.)

- How long will it last?
- Who is the chairman?

If you cannot get straightforward, punchy answers to these questions, then I suggest the meeting is likely to be a total waste of time, and you should refuse to go. (Clearly it is up to you to decide if internal politics require that you should attend anyway!)

ORGANIZING AND RUNNING MEETINGS

Here are nine steps towards better meetings.

1. Identify the objective(s) of the meeting.

- Identify the participants you need, and the chairman.
- Based on (1) and (2) build the agenda.

- Set a time constraint on each agenda item and on the meeting as a whole.
- Publish the objective(s), agenda, time constraints and indicate to each participant what preparation is required.
- Hold the meeting.
- The chairman drives the meeting towards the objective(s).
- Prepare an action list arising out of the meeting.

- Leave when the objective(s) has/have been met and the action list completed.

Chapter 20.

PRESENTATIONS

INTRODUCTION

INTRODUCTION

Your projects will invariably require you to give presentations: to your customer, to the project team, to your peers and to your superiors.

Some people are born presenters, one could almost call them actors. We have all seen them. They are totally authoritative on their subject. They come across as humorous, dramatic, somewhat informal,

relaxed, honest, caring, believing intensely in what they are saying. They tell their story in a punchy, effective way that is never boring, but instead grabs and holds our interest. They use plenty of eye-contact, and often employ audiovisual aids and props.

Others are less convincing. Perhaps they are smug, patronizing, overaggressive, too verbose and run on too long, too rehearsed, boring, too casual, dishonest or just unsure of their subject matter. Others still find presentations the most

terrifying ordeal imaginable; something which becomes apparent the minute they stand up and open their mouths.

It is not the intention of this chapter to teach presentation skills, i.e. how to present your material. There are enough media companies around today who can do that sort of thing, and who can give valuable advice on how to improve our effectiveness when speaking to an audience.

The aim here is to consider how best to structure your material

(what we present) for a presentation. Anybody, no matter how good or bad a speaker, can do what we recommend here; and the approach described will compensate for many speakers' shortcomings. Aside from this, it is up to the media companies to turn us all into Laurence Oliviers. Probably the best injunction ever on giving a presentation is:

- Tell 'em what you're gonna tell 'em

- Tell 'em
- Tell 'em what you told 'em

While it is hard to improve on this for the presentation itself, there is a lot you can do by way of preparation beforehand. Here are some guidelines I use.

- Identify the main messages you want to get across. There should be only a handful of these and they should be crystal clear. Also, you need to know whether you are merely

trying to explain and describe something or to persuade your audience about something.

- The trouble with crystal-clear messages is that they are also blunt. You may want to soften the messages a little by putting what I call *shading* on them. By this I mean putting some color into the black and white messages so that they appear happy/sad, optimistic/pessimistic, the

start of something big/the harbinger of doom, threatening/benevolent, light-hearted/serious, no problem/ big problem, or any one of a million other things. Blunt messages are just that. Shading means you anticipate the mood of your audience and deliver the blunt messages to maximum effect.

- If you are doing some kind of handout, structure it around these messages and their shadings.

- Rehearse your speech with a person or persons other than the intended audience. Explain the messages and shadings you are trying to put across and see if your presentation achieves those. Do this several times if necessary.
- Anticipate questions, both on your own and in rehearsal, and prepare answers which support your messages and shading.
- Questions can often take

you down paths you hadn't intended to go, and this sidetracking can cause you inadvertently to deliver messages and shadings you hadn't intended. Try to spot these potential sidetracks in rehearsal and anticipate how you will use them to reinforce your messages and shading.

- Give the presentation by following the adage above:
 - describe the layout of your presentation

- give the presentation, referring constantly to the layout, so that the audience can follow where you are; and deliver your messages with their shadings
- recap on the messages and shadings you delivered

Chapter 21.

SHORTENING PROJECTS USING ACCELERATED ANALYSIS AND DESIGN

[INTRODUCTION](#)

[WHAT TAKES THE TIME?](#)

[HOW TO CARRY OUT AN
AAD](#)

RISKS IN HOLDING AN AAD

PRACTICAL

CONSIDERATIONS

INTRODUCTION

In the 1970s there was a great swing towards the idea that more time spent getting the requirements and design right would result in:

- shorter time spent coding and testing
- fewer bugs
- an all-round better product

At the same time, tools started to appear which automated parts of the coding and testing process. Thus it seemed we had the best of all worlds — put the time and effort into the fun bit (requirements and design) and automate as much as possible of the time and labor-intensive stuff. As a result, conventional wisdom now has it that it is impossible to shorten the requirements and design phases of a project without seriously compromising the quality of the final product.

I believe this wisdom to be

fundamentally untrue; and the rest of this chapter shows a way of significantly shortening a project and, if anything, improving the quality of the finished product. The next section talks about what really consumes the time during the requirements and design phases of projects. The following section describes how to carry out an accelerated analysis and design (AAD) session. The fourth section talks about the risks involved and how to deal with them. The fifth section talks about some

of the practical considerations involved in carrying out AADs.

WHAT TAKES THE TIME?

Requirements and design phases are a bit like adultery. They typically consist of bouts of intense activity interspersed with periods of quite low levels of activity. It is not the intense activity that causes traditional requirements and design phases to be as long as they are. Instead it is things like:

- Large amounts of documentation to be produced; much of it

caused by the write, review, revise nature of traditional requirements and design approaches.

- Loss of continuity due to the put-down/pick-up nature of the traditional approach.
- Delay in getting responses and/or decisions.
- Sometimes somewhat unfocused effort either resulting in, or because of, low team spirit.

- Difficulty in getting the right people at the right time.
- Interruptions.
- People having other responsibilities ("Hey, I've got my own job to do!").
- The long time between reviews, walkthroughs or meetings means that it is possible for things to go a long way wrong before they are spotted.

- The natural overhead in traditional project management methods and structures.
- Meetings (and the difficulties of coordinating people's diaries).
- Getting things (decisions, documents) responded to, reviewed and approved.
- Reworking and refining (of documents, ideas etc.).

HOW TO CARRY OUT AN AAD

AAD addresses these problems in the following way. A one- or two-week-long meeting – an AAD session – is arranged. Ideally it will be held offsite. Invited to the meeting are all the decision-makers on a project, be they users and IS/IT people, in the case of an inhouse development, or engineering and marketing people in the case of developing a commercial

product. All the people who have a say have to attend.

A useful thing to do in advance of the AAD process is to have a sort of mini-seminar whose objective is to set both the AAD team's and upper management's expectations. This seminar would explain what the process is and what results can be expected. Having done this, there is preparation that needs to be done by each of the AAD team members. Each participant must independently write down:

- the name of the system/product being built/project
- a one-sentence description of what the system/product will do
- either three or four benefits of the system/product, or else three or four problems that it will solve

Apart from the attendees and, by the way, eight is about the maximum number you can

usefully have on an AAD two more people are required. The AAD session needs a leader, and it also needs what we call a technical scribe. The leader works from "eight till late"; the technical scribe works from after lunch until late. This is what they do.

On day one, the leader starts the AAD session with the group. The scribe may or may not be there, it doesn't really matter one way or the other. The group begins to put down the main requirements for the system or product. Typically

this is done at a board, flipchart, or (if your budget can rise to it) a photocopying whiteboard — one of the most wonderful inventions known to man. As the main requirements are hammered out, these are then further broken down, and this process will continue until the required level of detail is achieved.

I'm assuming here that you're doing your requirements and designs in English-language documents, on the basis that this would be the lowest common denominator for all

software development organizations. Thus, what you are doing during the AAD is taking a table of contents or template for a requirements or design document and filling in the blanks. If you've progressed from this and are using one of the structured methodologies or CASE, then AAD is even more effective when used with these approaches.

To continue with our AAD. The technical scribe shows up in the afternoon. Typically, this person is a senior designer level

person. The AAD session ends at say 5 p.m., and the scribe and the group leader take all the notes from the day and write as much of the requirements spec. as has been created. This document is then on each group member's desk when they arrive next morning.

The process begins again, using the fresh document as a starting point. Typically one would hope to have 95 percent of the requirements nailed down by the end of the second day, but it might take much longer than that. AAD sessions

are completely time-driven, and so you plan to do as much as you can in the time allowed, as opposed to having a fixed amount of work to do, and a variable amount of time in which to do it.

The days continue in this way with updated documentation being created each evening, and used as a starting point next morning. Results vary, but at the end of a two-week AAD session, you could expect to have a requirements spec. that was almost 100 percent correct (only small items of detail

should be in doubt, if any); and some proportion of the design of the system/product in place.

RISKS IN HOLDING AN AAD

In this section we talk about the risks involved in holding an AAD, and what, if anything, can be done to minimize them. Each risk is listed and then a possible action that could be taken.

- You end up with the wrong people. Say, for example, there's a senior decision-maker missing. Probably

the most serious thing that can happen. If this does happen to you, you need to kill the AAD session immediately. Careful preparation should reduce the risk of it happening.

- Group think. This is where the group gets so caught up in the AAD that ideas which aren't very good or even downright wrong are accepted and used. Try to pick attendees whose personalities are such that this probably wouldn't

happen.

- Foul up previously good relationship with users or marketing if the AAD doesn't deliver. No solution!
- Risk to one's own career if the AAD doesn't deliver. No solution. Good luck!
- Insufficient time in an AAD for subconscious problem solving. Sometimes we do our best work away from the office where we can often come up with highly

creative solutions to problems. An AAD doesn't really give us this luxury. This risk is offset to some extent by the synergistic effect of the AAD.

- Expectations not satisfied; particularly those of upper management. Clearly, these need to be set correctly. That's where the idea of the mini-seminar is very useful.
- The standards, or quality assurance, or quality

control department, view the procedure with suspicion, and slow down the procedure by insisting that all their standard stuff be done in addition to the output from the AAD.

- Some standards (e.g. document sign-offs) are there because they assume a traditional requirements and design life cycle. That is to say, many of them are based on a put-down/pick-up approach. For an AAD-type approach, the

standards should probably be revisited and altered to reflect the AAD way of working.

- Corporate or country culture. The hours and/or emotional intensity involved in an AAD may not suit every company. No solution!
- Part way into the AAD, the project is found to be too big so that the expected results cannot now be achieved in the timeframe.

Produce less of the deliverables or else take one chunk of the system down to the promised level of detail.

- Can't come to a consensus. A problem. Your best bet is that this effect can be offset by the feeling of people working together towards a common goal.
- Not much time in an AAD for tact. Yes, there isn't!
- Anti-climax/let-down after

the AAD is over. Create a post AAD-session job list so that there is natural follow-on from it.

- Requirements get missed. Yes, but this risk also exists in the traditional approach.

PRACTICAL CONSIDERATIONS

This section talks about some of the practical considerations involved in running an AAD. It does so under two headings:

- Who should come.
- What you will need.

Who should come

The leader Should be from outside the project, if not the organization.

Up to four analysts/designers These would be from the IS/IT department or software engineering organization as appropriate.

Up to four clients Generally one would expect to see more of these than analysts/designers. For the development of an IS/IT system, they would ideally represent different levels of potential user from

operational personnel up to management. For the development of a commercial product a mixture of marketing, sales and potential target customers would probably make sense.

The *project* leader If he is not already included among the group of analysts/ designers or clients.

The technical scribe A person with a good knowledge of analysis and design. Also, should be able to use a word processor!

What you will need

The room

- Ideally offsite
- Windows really important
- Walls to tape or pin to
- 24-hour availability
- No telephones

- Big tables to work at
- Comfortable chairs
- Refreshments. Away from the room is probably better, light lunches and no wine

Equipment

- White or blackboards. A whiteboard that photocopiers would be perfect

- 2 3 flipcharts and plenty of pens
- Overhead projector and screen
- Photocopier
- PC
- Laser printer
- Word processor
- Spreadsheet

- Project planning tool
- Software for producing slides/overheads
- Fax machine (possibly)

Miscellaneous

- Stapler
- Highlighters
- Paperclips

- Scissors
- Hole punch
- Pads and pens/pencils
- Sellotape

AFTERWORD

DELEGATION (OR THE
REAL JOY OF
MANAGEMENT)

DELEGATION (OR THE REAL JOY OF MANAGEMENT)

We only pass through this world once and we are limited in what we can do by the amount of waking hours available to us. We as project managers are in a unique and privileged position: we can use other people's time to accomplish what we want, thereby dramatically increasing the amount of time available to

us, and as a result, the things we can achieve.

Think about it. For each person on our team, we have that person's time to harness to our ends. It's like having our own lifetimes extended or even being given multiple lifetimes. We can only make truly effective use of this power if we delegate, passing down as much stuff as we can to our team members.

- Use the guidelines in [Chapter 17](#) to surround

yourself with the right people.

- Delegate as much stuff as you can away to them, following the guidelines in [Chapter 7](#).

Do this as fast as you can trust people to take it. If you do both of these things, you will find your days opening up with time that you can use for all those new things you always wanted to do. Then, and only then, will you experience the *real* joy of management.

APPENDICES

Appendix 1. ISO 9000 ESTIMATING PROCEDURE

INTRODUCTION

Section 1. WORK BREAKDOWN STRUCTURE, EFFORT, TASK DEPENDENCIES

Section 2. AVAILABILITY OF RESOURCES

Section 3. THE PROJECT MODEL

Section 4. BUILD IN CONTINGENCY

Section 5. IDENTIFY OPTIONS

Section 6. THE PREFERRED OPTION

Section 7. SAMPLE WBS

INTRODUCTION

This document describes an estimating procedure, based around Steps 1 – 5 of the Ten Steps, which could be used *by all personnel, not just project managers*, when making estimates. This procedure could form part of a series of project management procedures which could, in turn, go into a software development organization's quality manual. The procedure has seven sections.

[Section 1](#) shows how to develop a work breakdown structure (WBS) and to use this to make estimates of effort. This is the "demand" side of the project equation.

[Section 2](#) shows how to calculate the "supply" side, i.e. the availability of resources.

[Section 3](#) shows how to match the demand to the supply, thereby building a model of how your project might unfold.

[Section 4](#) shows you how to build in some contingency to

your plan.

[Section 5](#) shows you how to use this model to identify options and make sensible decisions about what can and cannot be achieved.

[Section 6](#) shows you what to do once a preferred option has been chosen.

[Section 7](#) contains a sample WBS.

1 WORK BREAKDOWN STRUCTURE, EFFORT, TASK DEPENDENCIES

Produce a WBS for the project

This is essentially a structured list of all the jobs that must be done to complete the project. The list should have a "folded map" format, that is:

- All the major milestones

shown you can get these immediately from Step 1 in the following section.

- As much detail as possible (i.e. down to the man day level) in the upcoming phase.
- As detailed as it can be thereafter.

Make the WBS as excruciatingly detailed as possible by milking every scrap of information you have for all that it's worth.

To help you to do this, a sample WBS is provided in [Section 7](#) of this appendix. This WBS follows a general purpose software development life cycle. The WBS is presented in such a way that you could add in your own life cycle elements and WBS checklist elements, gathered from your own experiences over time.

Steps to take ...

- 1. Write down all the jobs**

in your project and for each job identify:

- **which jobs it depends on**
- **which jobs depend on it.**

- Present this list in a top-down way with:
 - clear start and end points
 - a high-level overview, i.e. the (less than a dozen or so) top-level jobs that have

to be done to complete the project

- the milestones associated with these jobs
- wherever possible, these jobs broken down repeatedly until the lowest level of detail (i.e. the man-day level of detail) is achieved
- For the next 4-6 weeks, no job should be more than 5 man-days in size.

You can do this with whatever you normally use for writing, or alternatively and far better use a PC-based project planning tool.

- Guess and document the amount of effort (sometimes also called work) in each job. Note that it is effort, not elapsed time that we are looking for here. Nor is it the quantity called *duration* that PC-based project planning tools are so fond of. For each element write down the basis on which the guess was made,

e.g. the document has 12 sections, allow 1 hour per section, therefore 1.5 man-days; or "This is what one of these normally takes me" etc. If any data are available from previous projects, use them.

If you feel that you simply cannot guess certain jobs because they are unclear or unknown try first to break them down to lower levels of detail.

When you eventually feel you can go no further in this process, and if certain jobs are

still unclear or unknown which they will be then make and document an assumption. Finally, involve the people who will do the work in the estimating process. For that matter involve anyone whom you think might have a valid input.

Enter the guesses of effort for each job onto your list. Document the way each guess was arrived at. Note that this includes documenting all the assumptions you made.

When producing estimates with

an individual, the individual's normal working day should be used: no productivity factor should be used. Where it is not known who will work on a particular job, a productivity factor of 20 percent should be used. Thus a 10 man-day task actually becomes 12 days.

- Add up all the effort in all the jobs. This gives you the total amount of work in the project. Using manpower rates, it also gives you the project budget, if you need it. This is the "demand" side of your

project. All of this should be documented in the project plan.

2 AVAILABILITY OF RESOURCES

Staffing plan

Generate a staffing plan showing:

- Types of people required
- Number of each type required
- When the different people

will be required

Calculate the manpower available to the project

Typically, apart from your project, the members of your team will spend their time in a number of other areas:

- Definite other projects in which they are involved.
- Vaguer, support/maintenance/quality

request sorts of things which take up a definite chunk of time every week.

- Annual leave, public holidays.
- Unpredictable things which can't be anticipated. Sick leave comes under this heading.

The first three things can all be quantified, while some assumption can be made about the last point thereby quantifying it.

Steps to take ...

- 1. Work out for each person how much of their time (measured in days per week, hours per day or any other measure that means something to you) is being spent on these activities and hence how much time (measured in the same units) is available for your project.**

- Document each person's availability in your project plan and your PC-based project planning tool if you are using one.

3 THE PROJECT MODEL

Use the WBS and manpower availability to build a project model

Steps to take ...

- 1. Now match people to jobs. Wherever possible try to ensure that there is only one resource per task. Ensure that, as far**

as possible, every job has a human being's name as opposed to an organization or a "TBD" or "ANO" against it.

- Ensure that, as far as is possible, you have used each person's time available to the project exactly no more, no less.**
- If you are using a PC-based tool, use it to assign people to jobs. The tool will**

now draw a Gantt chart of your project for you. Among dozens, if not hundreds of useful things, the tool will show you your project's critical path.

- Failing this you will have to draw the Gantt chart yourself manually. If you are doing this, be warned that the manual approach will almost certainly**

introduce errors. The Gantt chart is a model of how your project may unfold over time.

4 BUILD IN CONTINGENCY

**Use the First Law of
Project Management to
build in a margin for error**

You have built a model of your project, either on your PC or on paper, and this model tells you how you believe your project will unfold once it gets on the road.

You could imagine, at this point, that you know things like the end date, the number of people required, the budget, and so on. However, you don't. The model you have created is exactly that: a model. It is a prediction of how the future may turn out.

Before you start making commitments based on the model you need to ask yourself one vital question. "What will I do if the model is wrong what will I do if it doesn't work out like I thought?"

A margin for error can be achieved by looking at the four parameters upon which the First Law of Project Management is based.

Just to remind ourselves, these are:

- Functionality what is to be delivered
- Delivery date when it is to be delivered
- Effort the amount of work involved in delivering

it

- Quality the quality of what is delivered

The First Law of Project Management states that there is a function which connects these four variables, and that this function is a constant, i.e.

Function (functionality, delivery date, effort, quality) = Constant

Thus, if one of these variables changes, the others change so that the overall equation

remains the same.

We are all familiar with this effect. Shorten the delivery date, for example, and (a) functionality must decrease, (b) effort must increase, or (c) quality must decrease.

Steps to take ...

- 1. To achieve a margin for error and a fallback position, examine your project using these four parameters as a basis.**

You can examine your project at three different levels, each requiring progressively more effort on your part. These levels are:

- overall (i.e. treating the entire project as one large job)**
- critical path jobs**
- all jobs.**

At each level, you examine the relevant job(s) and ask yourself

the following questions:

- **What happens if the job runs over? (Elapsed time)**

Can the job be shortened? (Elapsed time)

- **What happens if the job is bigger than we estimated? (Effort)**

What would be the effect of adding more people? (Effort)

- **Is this job doing something which is crucial to the project? If not, could it be put into a later delivery?
(Functionality)**
- **What happens if this job isn't done perfectly? (Quality)**

Based on the answers you should modify the estimates of effort and/or elapsed time for some or all jobs.

- Alternatively, you could add in a figure of, say, 15 percent to cover contingency. Add this to all effort estimates and this will automatically be reflected in extended deadlines and delivery dates.

5 IDENTIFY OPTIONS

Identify a number of different ways the project could be done

At this point it is quite likely that the end date and/or resources required and/or functionality which your project model is telling you is achievable, are not in line with what management or customers require.

If this is so, then you can use the model to see what is and what is not achievable.

1. Identify a number of options using the four parameters mentioned in Section 4:

- **What is to be delivered (functionality)**
- **When it is to be delivered (delivery date)**
- **The amount of work**

involved in delivering it (effort)

- **The quality of what is delivered (quality)**

A review of the model will answer questions like:

- **can the job be shortened? (Elapsed time)**
- **what would be the effect of adding more people? (Effort)**

- **is this job doing something which is crucial to the project? If not, could it be put into a later delivery?
(Functionality)**
- **what happens if this job isn't done perfectly? (Quality)**

to make creative decisions and identify options about how best to take the project forward.

- Again, having the model of your project on a PC enables you to do the "what-if" analyses described above very quickly. Based on what you discover, document the different versions of your project model.

Similarly, if you are doing this manually, modify your documentation to produce different versions.

6 THE PREFERRED OPTION

Introduction

The process of reaching a preferred option should be documented and the preferred option formally approved. Once this is done, a number of simple procedures should be carried out. The aim of these is to enable the project model to be used as "instrumentation" with which to guide the project.

Baseline the project model of your preferred option

Basically, this means taking a snapshot of your project model. Actuals will then be recorded against this so that the accuracy or otherwise of the model can be determined, and the model can be recalibrated where necessary.

All PC-based tools provide a facility for doing this. Alternatively you can merely

lay out your plan with:

- two columns for estimated schedule and effort these will have values in them
- two columns for actual schedule and effort these can be filled in as the project proceeds

Extract major milestones

The major project milestones should be extracted from the project model and explicitly

stated in the project plan.
These will serve:

- As useful intermediate targets to be met
- As an early warning system if the milestones aren't met

7 SAMPLE WBS

1 Product requirements phase

1.1 Product requirements document

- Research
- Write

- Circulate
- Individual review
- Review meeting
- Updates/changes to document
- Circulate again
- Second review
- Sign-off

1.2 End product requirements phase

2 Software requirements phase

2.1 Software requirements document

- Research
- Write

- Circulate
- Individual review
- Review meeting
- Updates/changes to document
- Circulate again
- Second review
- Sign-off

2.2 Software acceptance test plan

- Research
- Write
- Circulate
- Individual review
- Review meeting
- Updates/changes to document

- Circulate again
- Second review
- Sign-off

2.3 End software requirements phase

3 Architectural design phase

3.1 Architectural

design document

- Research
- Write
- Circulate
- Individual review
- Review meeting
- Updates/changes to document

- Circulate again
- Second review
- Sign-off

3.2 Software integration test plan

- Research
- Write
- Circulate

- Individual review
- Review meeting
- Updates/changes to document
- Circulate again
- Second review
- Sign-off

3.3 End architectural design phase

4 Detailed design phase

4.1 Detailed design document

- Research
- Write
- Circulate
- Individual review
- Review meeting

- Updates/changes to document
- Circulate again
- Second review
- Sign-off

4.2 Software unit test plan

- Research

- Write
- Circulate
- Individual review
- Review meeting
- Updates/changes to document
- Circulate again
- Second review
- Sign-off

4.3 End detailed design phase

Notes

- 1. During the first four phases, the cycle (circulate, individual review, review meeting, updates/changes to document) could be repeated more than once. You should state in your estimates how many iterations of this cycle you are assuming.**

- Research could be a significant project in its own right requiring much further detailed breakdown.

5 Coding phase

5.1 Produce code element

- Write code element
- Clean compile code element

- Link code element
- Walkthrough code element
 - Prepare for walkthrough
 - Attend walkthrough
 - Updates/changes to code
 - Sign-off on walkthrough
- Code element

documentation

5.2 End coding phase

6 Unit test phase

6.1 Unit test code

- Prepare test plan and test set
- Test code

- Make corrections to code
- Test code again
- Prepare unit test document

6.2 End unit test phase

Note

- 1. The cycle (test code, make corrections to code) could be repeated more than once. You**

should state in your estimates how many iterations of this cycle you are assuming.

7 Integration testing phase

7.1 Integration testing of code

- Any remaining preparation of test plan and test set, not already covered by

system integration test plan

- Test code
- Make corrections to code
- Test code again
- Prepare integration test document

7.2 End integration testing phase

Notes

- 1. The cycle (test code, make corrections to code) could be repeated more than once. You should state in your estimates how many iterations of this cycle you are assuming.**
- Both "Test code" and "Make corrections to code" could be significant projects in their own right requiring much further detailed breakdown.

8 System test phase

8.1 Execution of internal software acceptance test plan

8.2 End system test phase

9 Release phase

9.1 Installation

- Plans
- Activities
- Test
- Record results

9.2 Data conversion

- Plans
- Activities

- Test
- Record results

9.3 Reviews

9.4 Software release

9.5 End release phase

10 Operation and maintenance phase

10.1 Evaluation

10.2 Design reviews

**10.3 Support and
maintenance**

10.4 Audit

**10.5 End operation
and maintenance
phase**

11 Other possible WBS elements in life cycle

11.1 Training

- Familiarization by project personnel
- Training of project personnel
- User training

11.2 Recruitment

11.3 Test environment development

- Overhead in dealing with software development staff

11.4 Development support

- Database administration
- Development environment
- System build

11.5 Project management

See [Chapter 4](#) for a full breakdown of tasks. Allow 6 – 8 percent of total project effort, as described in [Chapter 4](#).

11.6 Configuration management

- Reviews
- Ongoing configuration management

11. 7 Documentation

- Reviews
- User (different types)

- Administrator
- Release notes
- Technical manuals, i.e. manuals describing how the software works
- Help texts
- Overhead in dealing with software development staff

11.8 Quality management/quality

plans

Appendix 2. STRUCTURED PROJECT MANAGEMENT (THE TEN STEPS) AND METHODOLOGIES

INTRODUCTION

INTRODUCTION

If you use no methodology at all in your project management, if all you use is your native cunning and survival skills, then over time it's possible that you will find yourself doing some of the things that make for successful projects. (I say it's possible, because you may not. Many people especially in software go their whole project management lives and never cotton on to what worked and

didn't work for them in the past. As a result they never know, when starting out on a project, whether or not it is likely to succeed. They live their lives in permanent suspense, agitation and worry. Sometimes things work out, sometimes they don't.)

If you go from this first situation to using a methodology you should probably see an improvement. Hopefully the methodology will force you to do some of the things that make for successful projects.

Ultimately, however, it's only by applying the Ten Steps that you will be doing all of the things that make projects successful. The Ten Steps are the "best" way to run a project. Thus, the Ten Steps can serve as a benchmark against which to compare other approaches.

I mention all of this as a way of explaining where methodologies fit into the scheme of things.

Methodologies cause you to do some of the things that make projects successful. A good methodology may cause you to

do a lot of them. No methodology that I'm aware of guarantees success. The Ten Steps do, solely because they were derived from observation of what makes other projects successful.

If you are already using a methodology, here would be a useful and interesting thing to do. Compare your methodology to the Ten Steps. Take each of your methodology's elements and see if that element has an analog in the Ten Steps. For example, your methodology may describe a way to do

estimating. The Ten Steps also describe a way (it's in [Chapter 2](#)). If you do this for all of the elements of your methodology, then for each element, three possibilities arise:

- It's in both your methodology and the Ten Steps
- It's in your methodology but not in the Ten Steps
- It's in the Ten Steps but not in your methodology

It's in both your methodology and the Ten Steps

This is the ideal situation. This particular element is essential to the planning and execution of your projects.

It's in your methodology but not in the Ten Steps

I would argue that you don't need it. It's window dressing. Your projects can get by

without it.

It's in the Ten Steps but not in your methodology

This is a gap in your methodology. In my experience, some of the things that usually crop up here, and they almost always do, are:

- The methodology does not stop you from committing to impossible missions

- The methodology doesn't require you to put contingency in the plan
- The methodology doesn't make clear how you should manage subcontractors or outsource vendors

You can see a full example of a comparison of the Ten Steps and a typical methodology in O'Connell (1999).

Appendix 3.

PROBABILITY OF SUCCESS INDICATOR

INTRODUCTION

CALCULATING A PROJECT'S
PSI

HOW TO CALCULATE THE
PSI

INTRODUCTION

We have described the current status of the probability of success indicator (PSI) and our research on it in Chapters 1–10. While there is still a lot of subjectivity in the scoring, our confidence in it has improved significantly in the three years it has been in use.

The original work I did on the PSI was described in [Chapter 20](#) of the first edition of this book. For those of you who are

interested in seeing how it has evolved, that material is retained here in this Appendix.

CALCULATING A PROJECT'S PSI

The PSI is a number lying in the range 0 – 100, which is assigned to projects using the scoring scheme given in the following section. The PSI can be calculated at any point in a project's life and gives a measure of how likely the project is to complete successfully. The PSI can be calculated in a variety of different ways. An individual (e.g. the project manager or an

external consultant) could do an assessment of the project and calculate the PSI using the rules below. Alternatively, a number of people involved in the projects (project leader, team members, management, customers) could be polled for their opinions (say, using a questionnaire) and the results averaged.

Who calculates the PSI, who gives input and how the input is gathered (questionnaire, interview, show of hands(!)) are all factors that can be considered if you are going to

use the PSI. Here I have chosen the quickest and simplest way, i.e. where one person (me!) does an analysis of a given project.

HOW TO CALCULATE THE PSI

The PSI is calculated by applying a score to each of the Ten Steps as they relate to the particular project. The maximum possible scores are as follows:

STEP	SCORE	TOTALS
1	20	
2	20	
3	10	
4	10	

5	10	70
6	10	
7	10	
8	10	
9	0	
10	0	30
<i>Grand total</i>		100

Just as an aside, notice how critical the planning stage is. I am firmly convinced that the success or failure of projects is determined in the first 10 percent of their lifetime. OK, back to the PSI. Here's how you determine each of the scores.

1 Goal {20}

Write down:

- A one-sentence description of the project [8]
- Three or four bulleted items on what constitutes project completion [6]
- A 2 – 3 page blurb which attempts to answer the questions on the visualization checklist (see p. 11) [6]

Having done this, score the project using the numbers in square brackets.

2 Job list {20}

- Is the job list up to date?
[4]
- Is it complete? Does it cover at least all the items in the checklist on p. 19?
[4]
- Are all the major milestones indicated and

well-defined? [6]

- Is it detailed to first milestone? [6]

Score using the numbers in square brackets.

3 One leader {10}

Questions like the following should help you to smoke out this one.

- Name the project leader.

- Is there a person who has the "fire in the belly" to get the project done?
- How many other projects does he or she lead?

Score as follows:

- | | |
|--------------------|----|
| ● 1 leader | 10 |
| ● 2 leaders | 4 |
| ● 0 or more than 2 | 1 |

4 Assign people {10}

Use Form 2 from Case Study 5 ([Chapter 4](#)) and the job list from 2 above to score all of the people-jobs in the job list. Add up all the scores and divide by the number of job-persons. Then depending on where the resulting number lies score from 10 as follows:

- | | | |
|--------|------|----|
| • 1.00 | 1.25 | 10 |
| • 1.25 | 4.49 | 4 |
| • 4.50 | 5.00 | 1 |

You could do this in two cuts, first at the person level (divide by the number of people), then

at the job-person level.

5 Margin for error {10}

The following tasks will help you to assess the margin for error:

- Write down the major risks
- Describe the fallback position
- Explain how, by differing from the final goal, this fallback position creates for

you a margin for error

You could do this at a number of levels:

- At the project level
- For the major milestones
- For those items on the critical path
- For each job-person

Lose 15 from your cumulative score if you have no margin for

error.

6 Leadership style {10}

Do the analysis required by Form 2 from Case Study 5 ([Chapter 4](#)). Compare with what's happening on the ground. Score out of 10.

7 Know what's going on {10}

Analyze the reporting and monitoring mechanisms in use,

and score out of 10. Lose points for no monitoring and controlling against the plan.

8 Tell people what's going on {10}

Analyze information dissemination mechanisms, for example, does everyone have an up-to-date copy of the plan; do they get it each time it changes? Lose points for no progress meetings and no progress reports.

9 Repeat 1 through 8 {No score}

10 The prize; The reckoning {No score}

EXAMPLES

Let's look at some examples. Two, which you know well by now, are Scott's and Amundsen's expeditions to the South Pole. Let's score them as our first two examples.

Scott

1. *Goal clearly defined? Not really. Only after he'd set out for the Pole did it actually become an objective. Half marks. (10)*

- *A list of jobs so that everybody knew their part. No. Hardly at all. People's roles were confused and kept changing. (6)*
- *One leader. No problem here. (10)*
- *Jobs assigned to people. As mentioned earlier, there was a problem with people's roles and what was expected of them. (4)*

- *Fallback position.* History proves there wasn't. (15)
- *An appropriate leadership style?* Scott had the one (autocratic) style that he used with everybody. I would score it low. (4)
- *Know what's going on.* Given that most plan was in Scott's head he had to have a reasonable knowledge of what was going on. Where he would have fallen down would have been in understanding undercurrents and morale-type issues. (6)
- *Tell people what's going on?* Hardly at all

This gives a profile as shown in [Table A.1](#):

Table A.1.

	STEP	POSSIBLE	ACTUAL	TOTAL
1	20		10	

2	20	6	
3	10	10	
4	10	4	
5	10	15	15
6	10	4	
7	10	6	
8	10	2	
9	0	0	
10	0	0	12
		<i>Grand total</i>	27

Amundsen

1. Goal clearly defined? Absolutely. (

- *A list of jobs so that everybody knew th*

part. Completely. But let's say he could have done a little better. (18)

- *One leader. No problem here. (10)*
- *Jobs assigned to people. Yes. See [Chapt](#) (or better still read Huntford (1993) if you it). (9)*
- *Fallback position. Yes they put on we remember? (10)*
- *An appropriate leadership style? Yes, Amundsen was very sensitive to his people*
- *Know what's going on. Absolutely. (8)*
- *Tell people what's going on? Yes. (8)*

This gives a profile as shown in the [Table A.](#)

Table A.2.

STEP		POSSIBLE	ACTUAL	TOT
1		20	20	
2		20	18	
3		10	10	
4		10	9	
5		10	10	67
6		10	9	
7		10	8	
8		10	8	
9		0	0	
10		0	0	25
			<i>Grand total</i>	92
<i>Project X1</i>				

This is a project I worked on about ten years ago. It was the one I mentioned in [Chapter](#) (Case Study 4) where there were two lead

1. *Goal clearly defined?* Sort of. There were no real specifications, but there were people on the project who had good knowledge of what the system was trying to deliver. Give it half marks, roughly. (9)

- *A list of jobs so that everybody knew their part.* We knew what we had to do from day to day; but did we know how that fitted into the bigger picture, and whether we were making progress? Definitely not. (6)

- *One leader.* No, there were two. (1)

- *Jobs assigned to people.* People were certainly assigned to jobs. However, given the job list may have been incomplete (see above) then we'd have to mark this low. (4)

- *Fallback position.* Well, I suppose there was a fallback because when the excrement hit the ventilator, management was able to reconfigure the project to deliver decreased functionality. (4)
- *An appropriate leadership style?* There were two leaders, one autocratic and opinionated and the other wishy-washy and a bit of a moan. (4)
- *Know what's going on.* See the comments above on job list and assignment of people
- *Tell people what's going on?* Somewhat

This gives a profile as shown in [Table A.3](#):

Table A.3.

STEP	POSSIBLE	ACTUAL	TOTAL
1	20	9	
2	20	6	

3	10	1	
4	10	4	
5	10	4	24
6	10	4	
7	10	4	
8	10	4	
9	0	0	
10	0	0	12
		<i>Grand total</i>	36

The Battle of the Somme

Here's one we mentioned earlier in the text opening of the Battle of the Somme in the War One.

1. *Goal clearly defined?* Yes, no doubt

about it. (20)

- *A list of jobs so that everybody knew their part.* Yes, dock 'em a couple of points on the basis that they might have done slightly better. (18)
- *One leader.* Yes. (10)
- *Jobs assigned to people.* Yes, done very well. (8)
- *Fallback position.* Nope. (15)
- *An appropriate leadership style?* Leaders of the World War One has been the subject of much controversy and literature. The prevalent style in the military of the time was "don't think, just do what you're told." I'll give them 10 points on the basis that more responsibility and initiative should have been passed down the line. (5)

- *Know what's going on.* The technology of the time made communications on the battlefield very, very difficult. Half marks. (5)
- *Tell people what's going on?* See earlier comments on "do what you're told." Half marks. (5)

This gives a profile as shown in [Table A.4](#):

Table A.4.

STEP	POSSIBLE	ACTUAL	TOTAL
1	20	20	
2	20	18	
3	10	10	
4	10	8	
5	10	15	41
6	10	5	

7	10	5	
8	10	5	
9	0	0	15
10	0	0	
		<i>Grand total</i>	56

Project X2

This is another project I once worked on.

- 1. Goal clearly defined? Yes, pretty well. We had a detailed spec. and descriptions of the functionality that would be in each release. (16)***

- A list of jobs so that everybody knew their part. No, we never really managed to stabilize our lists. We would get lists written, and then a short while later we would be told that*

"everything has changed," and the lists were no longer valid. (10)

- *One leader.* No. We had horrifying problems too numerous to mention with leadership. I gave me score 1 on the basis that we were in one or more than 2 category. (1)

- *Jobs assigned to people.* Well, because the job lists were in flux, so too were the job assignments. (3)

- *Fallback position.* Yes, we did. As the going got tough we were able to move functions from the first release into subsequent ones, a standard trick among software developers.

- *An appropriate leadership style?* Because of the problems described earlier, I eventually had to take control of this project and drive it through to a conclusion. Let me (modestly) score myself 7. (7)

- *Know what's going on.* By and large, yes did. (6)
- *Tell people what's going on?* We tried. (6)

This gives a profile as shown in [Table A.5](#):

Table A.5.

STEP POSSIBLE ACTUAL TOT

1	20	9	
1	20	16	
2	20	10	
3	10	1	
4	10	3	
5	10	7	37
6	10	7	
7	10	6	

8	10	6	
9	0	0	
10	0	0	19
		<i>Grand total</i>	56

Project X3

Here's another software project I worked on. The project manager is one of the best I know.

1. Goal clearly defined? Yes. (18)

- *A list of jobs so that everybody knew their part. Yes. (18)*
- *One leader. Yes. (10)*
- *People assigned to jobs. Yes. (8)*

- *Fallback position.* Yes, even though it was needed. (8)
- *An appropriate leadership style?* Yes. (8)
- *Know what's going on.* Yes. (8)
- *Tell people what's going on?* Yes. (8)

This gives a profile as shown in [Table A.6](#):

Table A.6.

STEP	POSSIBLE	ACTUAL	TOT
1	20	18	
2	20	18	
3	10	10	
4	10	8	
5	10	8	62

6	10	8	
7	10	8	
8	10	8	
9	0	0	
10	0	0	24
		<i>Grand total</i>	86

Operation Desert Storm

Clearly, I wasn't in on the planning of Operation Desert Storm. However, here's a man in the street's assessment of it.

1. *Goal clearly defined? Yes, with clinical accuracy. (20)*

- *A list of jobs so that everybody knew their part. Seems like. (18)*

- *One leader.* Yep, Mr Schwarzkopf. (10)
- *Jobs assigned to people.* Seems to have done well. (7)
- *Fallback position.* My guess is that there can't know for sure, though. (7)
- *An appropriate leadership style?* Don't s to come much better. (9)
- *Know what's going on.* We didn't, but presumably the powers that be did. (8)
- *Tell people what's going on?* Again, we c but I think those involved did. (8)

This gives a profile as shown in [Table A.7](#):

Table A.7.

STEP	POSSIBLE	ACTUAL	TOT
------	----------	--------	-----

1	20	20	
2	20	18	
3	10	10	
4	10	7	
5	10	7	62
6	10	9	
7	10	8	
8	10	8	
9	0	0	
10	0	0	25
		<i>Grand total</i>	87

Analysis

Trends

The projects I've just described are shown in tabular form in Figure A3.1. From the limited sample presented here and/or studied by us, a few things have become apparent.

- Any project with a planning phase score below about 40 looks dodgy.
- Any project with a total score below 60 looks dodgy.

- If you get a low score from the planning phase, then putting it charitably, you have limited scope to improve things subsequently. Putting it less charitably, if you come out of the planning phase with a low score, then it would appear that you can do what you like and it won't really make a lot of difference to the eventual outcome.

This last observation gives rise to my corollary to the famous

Brook's Law (Brooks, 1975) which states ... "Adding more people to a late project makes it later." My corollary states ... "Projects succeed or fail in the first 10 percent of their lifetimes."

Usage

What we have described here is a very simple risk analysis. It can be done at any point in a project's life. It is very much a garbage-in, garbage-out

exercise; if you tell it lies, it will tell you lies in return.

Limitations

At the moment the limitations of "taking the PSI challenge" (if I may call it that) include the following:

- It is based on a very small sample
- The scoring is somewhat coarse

- The scoring is somewhat subjective
- It has had limited application in real life
- Both the scale and scoring process need to be tuned and refined by reference to many other projects

That much having been said, we are now using the PSI on consultancy assignments and on training courses. Any projects we have studied to date have verified the accuracy

of our PSI scoring scheme.
We're looking forward to doing
more work in this area.

Figure A.9. PSI analysis table

PROBABILITY OF SUCCESS INDICATOR: ANALYSIS TABLE

	Marks available	Scott	X1	The Somme	X2	X3	Desert Storm	Amundsen
1. The goal	20	10	9	20	16	18	20	20
2. Job list	20	6	6	18	10	18	18	18
3. One leader	10	10	1	10	1	10	10	10
4. Assign jobs	10	4	4	8	3	8	7	9
5. Fallback position	10(70)	-15 15	4 24	-15 41	7 37	8 62	7 62	10 67
6. Leadership	10	4	4	5	7	8	9	9
7. Know what's going on	10	6	4	5	6	8	8	8
8. Tell people what's going on	10	2	4	5	6	8	8	8
9. Repeat steps 1 through 8	0	-	-	-	-	-	-	1
10. Prize; the reckoning	0(30)	-12	-12	-15	-19	-24	-25	-25
Total	100	27	36	56	56	86	87	92

Comments

- *Scott*. The worst project we have; ground zero on our scale
- *X1*. Scores very poorly as you would expect
- *The Somme*. Illustrates that in certain circumstances a margin for error can be a dealbreaker
- *X2*. Poor profile after planning just about saved by a strong implementation
- *X3*. Profile similar to

Amundsen; strong plan,
strong implementation

- *Desert Storm*. Almost as good as Amundsen, the best project we have

Rules of thumb

Anything coming out of the planning stage below 40 is very dicey. Failures will probably end up with scores of below 60.

Appendix 4. BASIC PRECEPTS AND GLOSSARY OF TERMS

INTRODUCTION

THE FOUR BIG ONES

ABBREVIATIONS AND
RULES OF THUMB

CRITICAL PATH

GLOSSARY OF GENERAL

PROJECT MANAGEMENT TERMS

INTRODUCTION

[Chapter 19](#) of the first edition of this book contained some basic information which I think is worth retaining here.

For something that is meant to be so complex, project management doesn't actually require much basic terminology. There are really four things you need to know about:

- Effort, also sometimes

called work

- Schedule, also sometimes called elapsed time
- Critical path
- Milestones

In addition, there are a whole heap of additional terms that are bandied about from time to time and these are presented in a glossary of terms at the end of this appendix.

THE FOUR BIG ONES

Let's look at the four big ones first.

Effort (work)

Effort is, quite simply, the amount of work in something. Effort is measured in units such as man-days, man-weeks or man-years. If a person digs a hole and it takes 5 days then the amount of work is 5 man-days. If 5 people dig a hole and

it takes them 1 day then that is also 5 man-days.

Schedule (elapsed time)

Schedule is the elapsed or calendar time that something is going to take. Schedule is measured in the ordinary calendar units of time: hour, day, month, year. In the previous example of digging the hole, the schedule in the first instance was 5 days; in the second it was 1 day.

Critical path

Critical path is a much misunderstood term. Critical path is the shortest time (i.e. the shortest schedule or elapsed time) in which something can be completed. In the examples which follow, you will see the significance of critical path.

Milestones

Milestones, as originally placed on roads, showed a traveller

how far she was from the start or end of her journey.

Milestones in projects do exactly the same. They "anchor" the project at particular points in time. They show how much of the project has been completed and how much is still to go.

ABBREVIATIONS AND RULES OF THUMB

You may also find the following of use:

8 man-hours = 1 man-day
(MH) (MD)

5 man-days = 1 man-week
 (MW)

Technically there are 4.3 man-days in a man-week, but 5 is easier to work with.

19 man-days = 1 man-month
(MM)

12 man-months = 1 man-year
(MY)

220 man-days = 1 man-year.

Technically, there are 228 man-days in a man-year, and I think this may be true in the US. Certainly in Europe it's closer to 220.



EXAMPLE 1

Pregnancy is an often used example of a project. For this, our four terms above are:

Effort	9 woman-months
Schedule	9 months approximately
Critical path	9 months (i.e. adding more women won't shorten the project) Doctors can identify particular

Milestones significant points in the growth of a child from conception to delivery.

EXAMPLE 2

Here's another project. It has six jobs in it as follows:

- Write document
- Circulate document
- Review document
- Revise and update document
- Circulate document again
- Sign-off document

The project ends when this particular document is signed off by the relevant parties.

Let's assume also that the following estimates, i.e. guesses made by us, apply to the six jobs. Each estimate is given as three numbers: effort in man-days (MD), schedule in elapsed time, number of people involved.

Write document	15.0 MD,	7.5 elapsed,	2 people
Circulate document	0.5 MD,	0.5 elapsed,	1 person
Review document	5.0 MD,	5.0 elapsed,	5 people
Revise and update document	4.0 MD,	2.0 elapsed,	2 people
Circulate			

document	0.5	0.5	1
again	MD,	elapsed,	person

Review

and sign-off	2.5	5.0	5
document	MD,	elapsed,	people

As always, we document the basis for these estimates:

1. *Write document* is 2 people working full time for a week and a half.

- *Circulate* is 1 person doing photocopying, faxing and mailing for half a day.
- *Review* is 5 people each spending a

day each reviewing the document and we give them a week to do it.

- *Revise* is the 2 people working full time, this time for 2 days.
- *Circulate again* is the same as the previous "Circulate."
- *Review* is the same 5 people taking half a day each to review and sign-off the document, and we give them another week in which to do it.

Thus:

Effort	27.5 MD
Schedule	20.5 elapsed
	20.5 elapsed
	(because the

Critical
path

tasks must follow one another sequentially. This is not always the case. Also each activity is on the critical path.)

Milestones

The completion of each of the six tasks represents one possible set of milestones in the project.

(The beginning
of each task is
an alternative
set.)

CRITICAL PATH

Critical path is important for a number of reasons.

- It tells you the shortest possible time in which the project will complete.
- It identifies those jobs which, if they are delayed, will result in a delay or slip in the project as a whole.
- It identifies those tasks

which must be shortened if the project is to be shortened.

- If a slip occurs on your project, the critical path identifies the jobs which must be worked on if the project is to be brought back on schedule.
- The critical path is the reason why adding more people to a project may not necessarily make the kind of difference we would expect. To illustrate this

let's see what happens if we add some people to our project above.

Nearly 70 percent of the effort on this project is taken up with two tasks: write document and revise document. If we were to add a person or even two people to these tasks our natural reaction would be to expect a substantial improvement on the schedule.

In practice, nothing could be further from the truth. Adding one person (i.e. increasing the

team by 50 percent) reduces the schedule to

$$5 + 0.5 + 5 + 1.5 + 0.5 + 5 = 17.5, \text{ a saving of 15 percent on the original;}$$

while adding another person and increasing the team by 100 percent reduces the schedule to

$$4 + 0.5 + 5 + 1 + 0.5 + 5 = 16.0, \text{ a saving of only 21 percent on the original.}$$

And note that adding more people assumes that they can

just slot in with no prior knowledge of the particular task, and be as productive as the people who were already there. Such an assumption is rarely the case in IT.

Indeed, there is an argument that says that in certain circumstances adding people to a project will not actually shorten the schedule at all. This argument is succinctly stated in Brook's Law (Brooks, 1975).

Adding people to a late project makes it later.

GLOSSARY OF GENERAL PROJECT MANAGEMENT TERMS

Glossary

Baseline schedule

A record of the task dates, duration, effort and costs, as originally scheduled.

Change management

The process by which a Project Plan is modified in the course of the project.

Critical path

A series of tasks, each of which must be completed on time to meet fixed task dates and/or the end date of the overall project.

Deadline

When a job must be complete by.

Deliverable

Something made, written, produced or created as a result of a job.

Dependency

A timing relationship between two tasks that determines the necessary sequence of events.

Detail task

A task that does not have any indented tasks beneath it.

Effort

How much work is in a job.

Elapsed time

How long a job will take.

Estimating

Guessing. Trying to predict the future. Doing this based on some previous knowledge or experience.

Filters

Filters are used to reduce the amount of data being displayed from the database. A filter acts on the data stream coming from the database. It does not touch the data in the database.

Gantt

Chart displaying the tasks on the project as bars on a timescale.

Goal

Think of the project as a journey; the goal is your destination. Every individual working on a project, the project manager and the project sponsor *must* be able to succinctly state the goal of the project. Only then will everyone be working in tandem. Also, there should be only *one* goal.

Job

Same as project or phase or activity or task.

Milestone

The date of a significant event, normally created as a task with zero duration.

Objective

Think of the project as a journey; an objective is one of the points you have to pass on the way to your destination. To put it another way, each job has an objective.

PERT

Project evaluation and

review technique. Flow chart of tasks that shows dependencies.

PERT Chart

A network whose nodes represent project jobs and their durations, and whose links represent relationships between pairs of jobs.

Phase, Activity, Task

These are all terms used to describe subcomponents into which a project can be broken down. In some organizations, these have specific meanings and relationships to one another. To avoid any conflict here, we use the term *job* as being synonymous with any of these three terms.

Predecessor

The earlier task in a dependency relationship between two tasks.

Project

Any endeavor can be considered as a project. In our terminology *project* and *job* are synonymous.

(Project) budget

The cost of all or part of a project.

Project control

Trying to keep what actually happens on a project in line with the Project Plan.

(Project) costing

Working out what a project will cost.

Project manager

Every project should have *one* and only *one* project manager who owns the project. This person is responsible for the project and will get the kudos when the project succeeds or will have her head

chopped off when the project fails. On large projects there may be a number of project managers, each with responsibility for a certain part of the project. Each project manager should be able to identify the boundaries of his project and the success factors associated with it.

Project monitoring

Checking how a project is proceeding against the Project Plan.

Project Plan

Your prediction of how you think the project will evolve. It is also the formal agreement between:

- the members of the team
- the team and the

project leader

- the project team and its customer.

Describes all aspects of the project identified during the planning process.

Reports

Reports are used primarily during the implementation stage of a project. They allow the user to calculate

totals and to cross-tabulate between tasks and resources.

Resource

The people, equipment and supplies used to complete tasks in a project.

Resource allocation

Assigning people (or other resources) to jobs.

Resource pool

This is the total of all resources in a project. The resource pool for a project can be maintained with the project or can be maintained in a separate project and thus shared by a number of projects.

Risk analysis

Trying to predict what might go wrong on a project and allowing yourself some room for maneuver.

Slack

The amount of time a task can be delayed without delaying a fixed task or the

end date of the overall project.

Success factors

What is the definition of a successful project?

- Within agreed timescale
- To agreed budget
- Supplies required functionality

- Supplies required quality

Successor

The later task in a dependency relationship between two tasks.

Summary task

A task that is a synopsis of

tasks indented beneath it in the outline structure. Also known as an outline task.

Task

An activity that has a defined start and end, and produces a definable result.

Variable cost

A cost defined as a price per unit of time.

WBS

Work breakdown structure.
A hierarchical organization of tasks using specific codes.

Appendix 5.

ADDITIONAL FORMS

THE ESTIMATING SCORE
CARD

THE CHANGE REQUEST
FORM

CHANGE REQUEST LOG

THE ESTIMATING SCORE CARD

Estimating score card	
Project	
Author	
Revision no.	
Date	

--

--

[illegible]

THE CHANGE REQUEST FORM

Change request form

Project name _____

Project number _____
Chg. req. number _____

Initiated by _____

Date _____

Description of change request

Reasons and perceived benefits

Priority: Essential to success of the project ☐
 Required for implementation ☐
 Can wait ☐

Affected tasks/deliverables

Estimate of cost and time

Approved ☐

Rejected ☐

Date _____

Date _____

Project manager

Project team reviewed

Yes ☐

No ☐

CHANGE REQUEST LOG

[illegible]

Appendix 6.

LEARNING

MICROSOFT

PROJECT 2000

INTRODUCTION

MODULE 1: BASICS OF PROJECT MANAGEMENT

MODULE 2: GETTING STARTED WITH MS PROJECT 2000

MODULE 3: MENUS OF MS

PROJECT 2000

MODULE 4: SETTING DEFAULTS FOR YOUR PROJECT

MODULE 5: CREATING TASKS

MODULE 6: ENTERING TASK DURATIONS

MODULE 7: USING HELP

MODULE 8: USING VIEWS

MODULE 9: SETTING TASK DEPENDENCIES

MODULE 10: USING ORGANIZATON AND PROJECT CALENDARS

MODULE 11: OUTLINING TASKS

MODULE 12: PRINTING VIEWS

MODULE 13: ASSIGNING RESOURCES

MODULE 14: OPTIMIZING THE SCHEDULE

MODULE 15: RESOURCE LEVELING

MODULE 16: USING THE BASELINE

MODULE 17: UPDATING TO REFLECT ACTUAL PROGRESS

MODULE 18: MULTIPLE PROJECTS

MODULE 19: USING REPORTS

INTRODUCTION

The purpose of this appendix is to present the minimum functionality of MS Project 2000 required by a project manager to assist the running of a project. It is based on a course run by ETP and uses MS Project 2000 to plan and schedule an exercise project: the construction of a golf course.

The notes follow the standard steps required during the

planning and execution of a project:

- Creating tasks
- Editing and moving tasks in the plan
- Setting task durations
- Setting dependencies between tasks
- Formatting and presenting the plan

- Assigning resources to tasks
- Saving the baseline

The user of these notes is assumed to have a basic understanding of how to use the MS Windows operating system.

MODULE 1: BASICS OF PROJECT MANAGEMENT

Overview

The basics of project management module reiterates Steps 1 and 3 of ETP's structured project management methodology. This module also briefly discusses key project management topics.

Objectives

The participant will understand that a project must have:

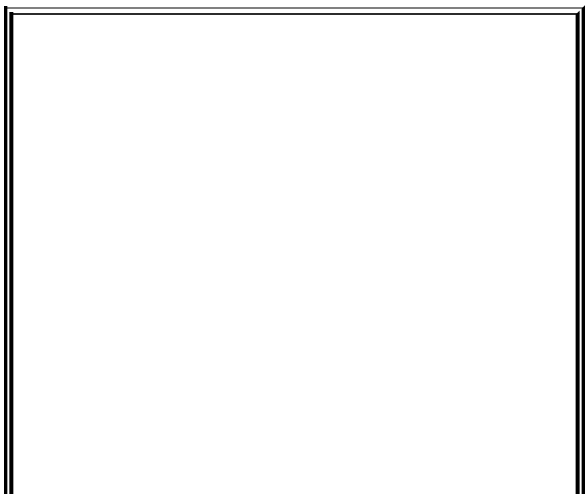
- A clear goal.
- Only one identified leader.

Preparation

Review the contents of this module. Be familiar with the terms in [Appendix 4](#) and be able to explain what each term is and how it is used when managing projects.

Presentation

MS Project is a tool that you can use to help with the structured project management methodology.



ACTIVITY

If the participants have not done this exercise before in a Structured Project Management Course, have them do the following:

EXERCISE 1.1 • **LEVEL: BASIC**

In your notepad:

- List the name of your project
- List the goal of your project
- List the project manager
- List the success factors

Dialog

Remind the participants of Step 1,

visualize the goal. Reiterate the four parameters of a project from the First Law of Project Management (note the success factors that define a successful project).

Also remind them of Step 3, there must be one leader. The project manager will be the one responsible for preparing or supervising the preparation of the MS Project Plan. The project manager will need to understand all the elements of the MS Project Plan so that he can come up with the options for going forward with the project.

Finally, explain to the participants that MS Project has been developed using specific terms which represent the various data in the database that each MS Project file uses. Have them turn to [Appendix 4](#) and go through each term, explaining how it is

represented in MS Project (i.e. draw a picture of what it looks like, show and tell).

MODULE 2: GETTING STARTED WITH MS PROJECT 2000

Overview

The getting started with MS Project 2000 module introduces the major features of the MS Project 2000 product.

Objectives

The participant will:

- Understand the different components that comprise MS Project 2000.
- Know the range of toolbars.
- Be comfortable with the screen displays.

Preparation

Review the contents of this module and know what each toolbar is called and what

components are represented in each.

Presentation

Review the toolbars with the course participants.



ACTIVITY

Have the participants follow as you explain the toolbars to them.

Dialog

Explain how MS Project is a powerful calculator and should be used to assist you managing the project.

Note that it cannot manage the project without input from a human being who understands the implications of manipulating the data.

It does not manage the project for you.

If you are a poor project manager, then using MS Project 2000 will probably make you worse. If you are a good project manager then using MS Project 2000 will give you greater control of your project and will increase your ability to communicate

the project status.

Remind the participants that, as with other Microsoft products, MS Project 2000 has a number of ways of doing the same thing. They will be shown some of those variations in this appendix. Another point to note about MS Project 2000 is that it has been designed to cater for every possible project type and every possible project management style. It does not force you into doing things its way. You can develop and use your own style and MS Project 2000 will fit your way of managing projects. The downside of this, however, is that MS Project 2000 may at first seem rather complicated. There are at least two ways of doing everything in MS Project 2000, and sometimes six or seven. In this course we shall use a number of different ways, and in your

own projects you can choose whichever suits you best.

Take the participants through each of the toolbars, etc., shown on the top of their screen. Some facilities will have equipment to display the instructor's desktop on a screen for all to see. In this case, have the participants look at the screen. Otherwise, explain each of the toolbars and their contents verbally to the class. The standard MS Project 2000 screen contains a number of different areas:

This is a
standard
Windows
application
menu. It

Menu line

allows access to all underlying functionality. The menu is in the same format as in other Microsoft Office products. This is simply a shorthand way of accessing frequently

Standard toolbar

used menu options. The first ten buttons, or icons, are identical for all the Office products. A feature, Tooltips, is available which indicates the function of each icon. To activate this just pause

Formatting toolbar

the cursor on a icon.

This supplies additional icons to the toolbar to help outline your project and format your text.

The number and content of the toolbars can be changed.

Entry bar

This allows editing of

Status bar

your input.
This line is
displayed on
the bottom
of the
screen.

Display options

You can
control
whether or
not you wish
the toolbar,
the ribbon
and the
status bar to
be
displayed.

You do this
using the
Options or
Customize
commands
within the
Tools menu.

MODULE 3: MENUS OF MS PROJECT 2000

Overview

The menus of MS Project 2000 module introduces the major components of the nine MS Project 2000 menus and the commands in each.

Objectives

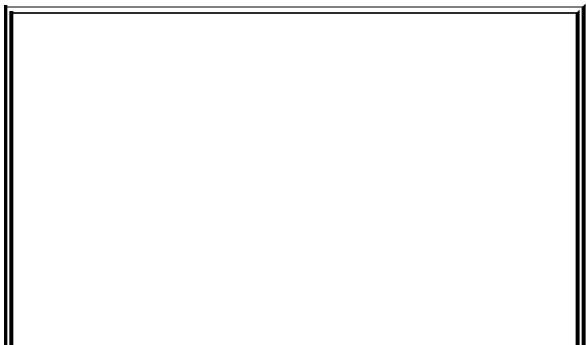
The participant will:

- Become familiar with the nine menus of MS Project 2000.
- Be aware of the commands in each menu.
- Understand that where a command shows an icon beside it, that icon will be displayed on another toolbar and is a shortcut to doing the same command.

Preparation

Review the contents of this module and be able to explain each of the nine (9) menus and the commands in each.

Presentation



ACTIVITY

Have the participants follow as you explain the menus and the commands to them.

Dialog

Demonstrate each of the menus in MS Project and note the commands in each. Also explain that where the participant sees an icon or picture next to a command, this means that the icon is also located on a toolbar and that using the icon is a shortcut to doing the same command.

This menu provides the file save and retrieve and the print functionality.

File

It also provides functionality to view the project properties.

This menu provides the cutting and pasting functionality; as

Edit well as editing tasks and resources and identifying task dependencies.

Remember to explain that views change the display that the user sees and that the user can print the image displayed.

Views

Views are a most powerful feature of MS Project 2000. This menu provides a number of predefined views plus the ability to define new ones. It also provides a number of predefined tables plus the ability to define new ones. It provides access to the

report gallery
and access to
different
toolbars.

The different
views provide
many ways of
looking at the
one underlying
database in MS
Project 2000.
Either a single
view or a
combination
(two) view can
be displayed.
Each view has a

name and can be defined by the user.

Also note that the Gantt chart is the default view for MS Project. The Gantt chart is by far the most familiar to all the participants, and it is the easiest way to show what has to be done and when.

The Gantt chart view is the default view visible on startup. This default can be changed if required.

**Views
(cont'd)**

The project database holds data on three areas of the project:

- Tasks
- Resources
- Assignments of resources to tasks

Some of this data (e.g. task names and durations) is input by the user and the rest of this

data (e.g.
total project
duration) is
calculated by
MS Project
2000.

To continue with the menus...

This menu
provides the
ability to insert
new tasks and
recurring tasks.
It also provides
the

Insert

functionality to insert drawings or objects into a file.

This provides formatting functionality for the text in a file and also provides formatting for specific types of information. In addition, it allows you to create predefined

Format

styles for text
and Gantt chart
bars.

Tools

This provides a
large amount of
miscellaneous,
but essential,
functionality.

The Project menu is new in MS
Project 2000 and is not in previous
versions of MS Project.

This menu
allows you to
see summary
project
information

Project

such as start
and end dates,
budget
consumed
against plan
and other task
and project
specific
information. It
also provides
the sorting and
filtering
functionality.

Demonstrate the ability to split the screen and indicate that there will be an exercise on this later in this appendix.

This is the standard Windows application menu which allows you to display a number of windows together. This is useful if you are working on a number of projects at once. This menu also controls the

Window

displaying of
two views
together.

Finally, explain that Help will be
looked at more extensively later in
the course (see [Module 7](#)).

This provides
comprehensive
help accessed via
the Windows
hypertext
functionality. It
also provides
access to the cue
card system from
where you can

Help

access the tutorial. Note that you can also use the F1 key to get context sensitive Help.

Indicate that both menus and toolbars can be altered to display more or less information. Details on how to do this are available in MS Project 2000's User Guide; however, for a more advanced class, there are exercises later in this appendix on how to do this. The content of some of the menus can change depending on the context from which they are accessed. Note also that you can change the layout of the menus if required and can also change the number and meaning of the

commands in each one.

MODULE 4: SETTING DEFAULTS FOR YOUR PROJECT

Overview

This module introduces how to set defaults for your current project and also for all future projects.

Objectives

The participant will:

- Learn how to set defaults for frequently used fields (e.g. date, schedule, view).
- Know where the default settings are located.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise and whether the level is suitable for basic or advanced users.



ACTIVITY

4.1 How to set defaults for displaying the date when you open a new MS Project file.

EXERCISE 4.1 • **LEVEL: BASIC** • **SET DATE DEFAULTS**

- Pick Tools menu. Choose the Options command.
- Select the View tab.
- Choose appropriate Date format.
- Click on the OK button.

Dialog

Explain that defaults can be used to

tailor information to suit the user. The customizable settings are accessed from the Options command on the Tools menu. For additional ways to customize information, look at the User Manual.

ACTIVITY

4.2 How to set defaults for tasks (e.g. requiring a specific amount of resources to complete, a fixed amount of time or a set amount of work).

EXERCISE 4.2 • **LEVEL: ADVANCED** • **SET SCHEDULE DEFAULTS**

- Pick Tools menu. Choose the Options command.
- Select the Schedule tab.
- Default task type = fixed units.

Dialog

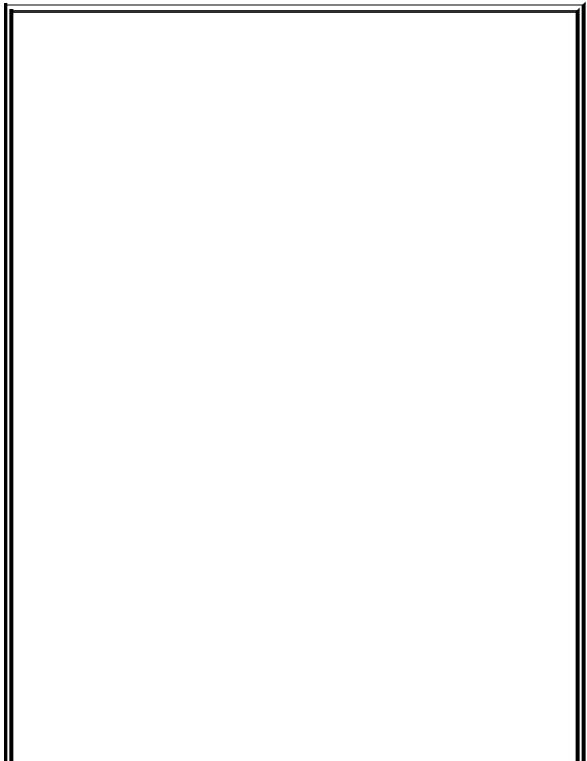
Explain the formula that MS Project uses to calculate duration.

Tasks can be either fixed units, fixed work or fixed duration. To understand this we need to analyze the formula MS Project 2000 uses to calculate duration. In its simplest form, the formula is

$$\text{work} \div \text{units} = \text{duration}.$$

We can fix one of these by default, amend another and the third variable will be the calculated result.

Duration formula



ACTIVITY

4.3 How to use the context sensitive help button to find out the definition of setting defaults for tasks.

EXERCISE 4.3 • **LEVEL: ADVANCED** • **ACCESSING HELP** **INFORMATION**

- Click the context sensitive help button (? at the top right corner of the dialog box) and read the description of this field.
- The cursor will change to a ?.
- Point at the Date Format and Click.
- When you have read the display, click to remove it.

Dialog

Explain that there is a separate module to examine the Help function.

MODULE 5: CREATING TASKS

Overview

The creating tasks module introduces how to insert, move and edit tasks in a project. It also demonstrates how to set up background information for the project.

Objectives

The participant will:

- Be able to put tasks in sequential order.
- To edit task names and remove redundant tasks.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims for each exercise and whether the level is suitable for basic or advanced users.



ACTIVITY

5.1 How to set the start date when you open a new MS Project 2000 file and how to input a title and other information relevant to the project.

EXERCISE 5.1 • LEVEL: BASIC • SET UP INITIAL PROJECT INFORMATION

- Pick Project menu.
- Pick Project Information command.
- Set Start Date = "April 12, 2000." Click OK button.
- Pick File menu.
- Pick Properties command.

- Pick Summary tab.
- Title = "Golf Course Construction."
- Manager = (your name).
- Comments = "This project covers the building of a new golf course."
- Click the OK button to accept the input.

Dialog

Explain that the only date that should be specified for a project is its start date. MS Project 2000 will calculate all other dates, based on the relationships that are input into the file. We then identify each task in the

project and how long it will take to complete before identifying any relationships between tasks.

The first thing we wish to do is to set up some general information about the project. We use the summary information dialog box to do this.

Within the dialog box we use the tab key to move from field to field. Use Shift+tab to move backwards.

Alternatively, use the mouse. During this course we will use as our main exercise the building of a golf course.

Make the following points when inputting a task:

1. Can your task be written or broken down into subtasks?

- Have you noted a clear start?

- Have you used abbreviations?
- Is your language clear?
- Have you indicated third party interests?
- Use generic descriptions?
- Have you noted your final task/project goal?

ACTIVITY

5.2 How to enter the name of the task in the Gantt chart.

EXERCISE 5.2 • LEVEL: BASIC • ENTER THE TASK NAMES USING THE GANTT CHART

- Use mouse, tab or arrow keys to select the task name cell of Row 1.
- Enter "Design accepted," hit CR.
- Note that the next row is selected.
- Enter "Clear land," hit CR.
- Enter "Obtain permission," hit CR.

Dialog

Explain that the Gantt chart is one of the ways of entering task details. It is also the only method that will be demonstrated in this appendix. Note the fact that when a task is entered into the MS Project 2000 file, a series of data is automatically created for that task.

There are many ways in MS Project 2000 to create task details. In this course we will normally use the Gantt

Creating tasks

chart as the first method of entering task details.

Also note that it is quicker and easier to enter the task list first and then go back to alter the durations for each task. When you enter a task and hit the CR or enter button, the task is accepted by the program, assigned a duration of 1 day and the next task name under the one entered is highlighted.

The quickest way to enter tasks is to use the Gantt chart and

enter the
names and
durations
separately.

Entering tasks

Type the task
name in the
Name column
and use CR.
When the task
names are
entered,
repeat the
process for
the duration
column.

ACTIVITY

5.3 How to correct errors by using the delete, clear and undo functions in MS Project 2000.

EXERCISE 5.3 • **LEVEL: BASIC • DELETE, CLEAR AND UNDO**

- Click in task name column of Task 3, "Obtain permission."
- Pick Edit menu and pick Clear, All.

This only deletes the task name. We didn't mean to do that so we can undo what we have just done.

- Pick Edit menu and pick Undo Clear.

- Pick Edit menu and pick Delete task.

This deletes the complete entry. However, we now realize we needed that task after all.

- Pick Edit menu and pick Undo Delete.
- Press the Del key on the keyboard.

This also deletes the complete entry. We now realize we needed it after all so use Undo to retrieve it.

Dialog

Explain that, as in other Microsoft products, there is the ability to undo or alter information if an error is

made.

Correcting errors

MS Project
2000 has a
powerful
undo
feature if
you make a
mistake and
wish to
backtrack
on
something
you have
just done.

ACTIVITY

5.4 How to move a task, either by using the cut and paste method or by dragging it.

EXERCISE 5.4 • **LEVEL: BASIC** • **MOVE A TASK**

We want the task "Obtain permission" to come before the task "Clear land" in the list.

Method 1

- Select the complete task "Obtain permission" by clicking in its ID column. This is the column with the number in it.
- Pick Edit menu, pick Cut Task.
- Click on "Clear land."

- Pick Edit menu, pick Paste Task.

Method 2

- Select task "Obtain permission" by clicking in its ID column.
- Click cut icon.
- Click on "Design accepted."
- Click on paste icon.

Method 3

- Select task "Obtain permission" by clicking in its ID column.
- Point to the ID number cell, hold down the mouse button and drag the complete row to

where you wish to place it.

Dialog

Remind the participants that when they write down the list of tasks, sometimes they wish to alter the order of that list. Re-ordering tasks can be done by either cutting and pasting information or by dragging information.

MS Project
2000 does not
worry about
the order of the
tasks in the list
but sometimes
we want to
move tasks to

Moving

tasks

make the order
more
meaningful for
ourselves.

There are a
number of ways
of doing this.

ACTIVITY

5.5 How to edit a task name by using the entry bar toolbar.

EXERCISE 5.5 • **LEVEL: BASIC** • **EDITING A TASK NAME**

- Select the task name field "Obtain permission."
- The name appears in the entry bar.
- Click between the words "Obtain" and "permission" in the entry bar.
- Insert the word "planning."
- Click on the Enter box (the tick), or hit CR.

Dialog

The entry bar allows us to alter the text that we have previously entered.

**Editing
an
entry** We can alter
any text by
selecting it and
then using the
entry bar to do
any edits we
wish.

MODULE 6: ENTERING TASK DURATIONS

Overview

The entering task durations module introduces how to set durations for tasks. This is the first estimate of the elapsed (scheduled) time to complete a task. It is only the first estimate because duration is calculated properly when a resource and an amount of effort are associated with a

task.

Objectives

The participant will:

- Be able to specify durations for tasks.
- Be able to set milestones showing progress checkpoints during the project.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise and whether the level is suitable for basic or advanced users.



ACTIVITY

6.1 How to enter durations for tasks established in the Task Name column.

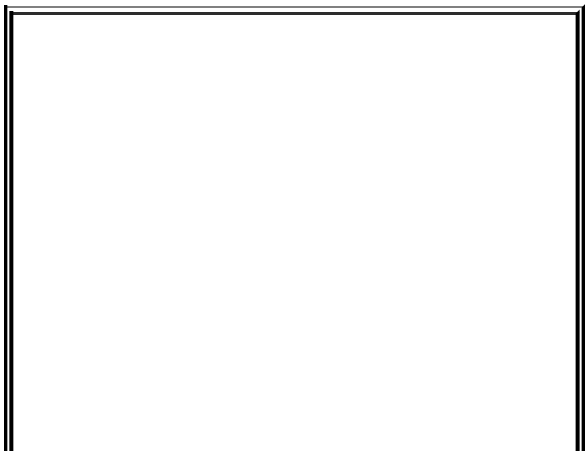
EXERCISE 6.1 • **LEVEL: BASIC** • **ENTER DURATIONS FOR CURRENT TASKS**

- Open **Golf1 B.**
- Select duration field for "Clear land."
- Enter "4w" and hit CR.

Dialog

Explain that the duration field is represented on the calendar side of the Gantt chart by a bar, and that this bar will change when the time

indicated in the duration column changes. Duration is the time needed to complete a task. A task's duration can be expressed in minutes (m), hours (h), days (d), or weeks (w). If no letter is specified, d is assumed. Non-working periods, for example weekends or public holidays, are not included in a task's duration.



ACTIVITY

6.2 How to use the Fill Down command.

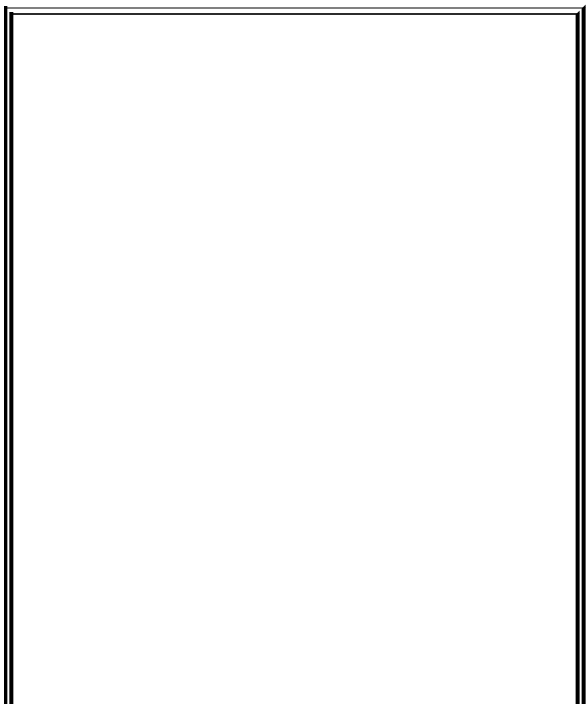
EXERCISE 6.2 • **LEVEL: BASIC** • **USING FILL DOWN**

- Select duration field for "Clear land."
- Hold down the mouse button and select the next 5 task durations.
- Pick Edit menu. Choose the Fill Down command.

Dialog

Explain that one can change the duration of a number of tasks at the

same time by using a command called Fill Down.



ACTIVITY

6.3 How to establish a Milestone in the Gantt Chart.

EXERCISE 6.3 • **LEVEL: BASIC** • **SET A MILESTONE**

- Select duration field for "Design accepted."
- Enter "0" and hit CR.

Dialog

Explain that a milestone is represented by a black diamond shape with a date beside it. All milestones have no duration associated with them as they represent significant events or deliverables in the project.

A milestone
is an
important
checkpoint in
the

Milestone schedule.

You specify a
milestone by
setting the
task duration
to zero.

ACTIVITY

6.4 How to open the task information box in the MS Project 2000 file.

EXERCISE 6.4 • **LEVEL: BASIC** • **ACCESS THE TASK INFORMATION BOX**

Method 1

- Pick Project menu.
- Choose Task Information command.
- Click OK or Cancel to close the dialog box.

Method 2

- Click on task information icon on toolbar.
- Click OK or Cancel to close the dialog box.

Method 3

- Double click on the Task name.
- Click OK or Cancel to close the dialog box.

Dialog

Explain that the task information box is a useful tool that provides information about each task. The information is stored in the database that sits underneath the MS Project 2000 file. For more information about the contents of the task

information box, use the context sensitive help (?), which is available on each dialog box to explain the contents in more detail.

ACTIVITY

6.5 How to use the task information box to change a number of tasks at the same time.

EXERCISE 6.5 • **LEVEL: BASIC • MAKE CHANGES TO MULTIPLE TASKS**

- Select Tasks 3, 5 and 6.
- Display task information box.
- Set duration field to "2w."
- Click OK.

Dialog

Explain that data entered into the task information box will be reflected

in the Gantt chart. This way, you are not limited to making changes to the project plan by using the table part of the Gantt chart.

To change a number of items at the same time, highlight the items by using the click and drag or the click and shift key technique. To highlight items that are not adjacent to one another, use the click and control key technique.

Shortcut for selecting multiple items (e.g. tasks) in a Windows environment

If items are beside one another, point to first item, hold down mouse button and drag to last;

or click on first item, point to last item, hold down shift key on keyboard and click with mouse.

If items are separated, click on first item with mouse, point to next item, hold down Ctrl key on keyboard and click again with the mouse button.

MODULE 7: USING HELP

Overview

The using Help module introduces how to find answers to queries in the most efficient manner.

Objectives

The participant will:

- Become familiar with the help functions in MS Project 2000.
- Be introduced to all the functionality that the Help menu offers.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise and whether the level is suitable for basic or advanced users.



ACTIVITY

7.1 How to access context sensitive help.

EXERCISE 7.1 • **LEVEL: BASIC • DISPLAY HELP ON INSERTING TASKS**

- Click the Help menu.
- Choose the Microsoft Word Help command.
- A Help topics window will appear requesting that you type in a request.
- Type *insert* a task.
- Click the Search button.

- A list of general topics will appear.
- Select the topic *Enter a task into a project* and read the information provided.
- When you have finished reading, click either the minimize or the close icon on the top right of the window to hide it, or the Help topics button to return to Help.

Dialog

Explain that this command is called context sensitive help and can be accessed in a variety of ways by using the Help menu as this exercise shows, by clicking on the balloon with the ? in it (see your toolbar), or by clicking on the F1 key.

A number of the help windows will have a hypertext phrase called *See also*. Use this to find out more about the topic you are looking into. MS Project 2000 uses a type of help known as *Office on the Web*. It has an extremely large volume of online reference information which can be accessed in a variety of ways depending on your need. The most frequent method you will use is the context sensitive help, accessed by the F1 key. This can be particularly useful when you get a warning or error message. On most dialog boxes there is a help button. Use this frequently.

ACTIVITY

7.2 How to use the Index tab in MS Project 2000 Help.

EXERCISE 7.2 • **LEVEL: BASIC** • **DISPLAY HELP ON CREATING DEPENDENCIES**

- Click the Help menu.
- Choose the Contents & Index Topics command.
- Click the Index tab.
- Type *create* and note that the index skips to all topics beginning with that word.
- Find create dependencies.

- Click on the Display button.
- Select create a task link.
- Click the Display button.
- Read the information presented in the box.
- Also read the hyperlink topics which are shown as words in a different color to the text and are underlined, as well as the topics highlighted with a double arrow.

Dialog

Explain that the index feature is much like an index at the back of a book. All the topics are listed in alphabetical order and you do not need to use

the CR or enter keys to go to any place in the index. A common way into the help system if you know the subject on which you wish to get help is to use the Help command on the menu and then use the Index command.

ACTIVITY

7.3 How to use the Contents tab in MS Project 2000 Help.

EXERCISE 7.3 • **LEVEL: BASIC** • **FIND OUT ABOUT TASK DURATIONS**

- Click the Help menu.
- Choose the Contents & Index Topics command.
- Click the Contents tab.
- Select the Creating a Project book and click the Open button.
- Select the Starting a Project book and click the Open button.

- Select Setting the length of a task option and click the Display button.
- Select each of the three topics (hyperlinks) in turn, and read.

Dialog

Explain that the contents feature is like a table of contents at the beginning of a book. Each chapter is represented by a purple book and has a series of sub-chapters or sections in each. The index is extremely large and you may sometimes find it difficult to find the exact topic you are looking for. Another way of looking at Help is to imagine you are looking at a large set of books. This is the way the contents are organized. Assume for example that you want to find out

more about task durations.

ACTIVITY

7.4 How to go to a cue card in MS Project.

EXERCISE 7.4 • LEVEL: BASIC • USE CUE CARD TO LEARN HOW TO CHANGE TASK DURATION

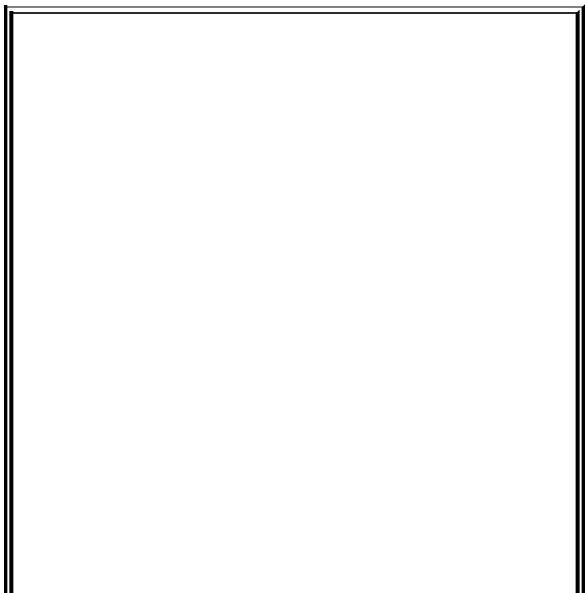
- Click the Help menu.
- Choose the Contents & Index Topics command.
- Click the Contents tab.
- Select the Creating a Project book and click the Open button.
- Select the Starting a Project book and click the Open button.

- Select Change a task duration and click the Display button.
- Read the contents and then click on the >> at the end of the dialog.

Dialog

Explain that Help uses both dialog boxes and cue cards to assist the user with MS Project. Whereas the dialog boxes explain the topic and have hypertext links (which are in green and underlined text) to other text, the cue cards are a step-by-step approach that the user can use to help follow through on a procedure. The index is one method of navigating the help system. An alternative method of navigating is by using cue cards. A cue card is a step-by-step procedure that you can

follow when you need to undertake a certain task. You can display a cue card while you work in MS Project 2000 and it stays on the screen until you minimize or close it.



ACTIVITY

7.5 How to go to the MS Project map in Help.

EXERCISE 7.5 • **LEVEL: BASIC • MS PROJECT MAP**

- Click the Help menu.
- Choose the Getting Started, Microsoft 101: Fundamentals command.
- Click on "Show me a map to Microsoft Project."
- Click on 1 and read descriptions and instructions.
- Click on Back button and repeat as necessary.

Dialog

Explain that the MS Project map shows the steps to follow when using MS Project. This can be used to remind users what to do next.

ACTIVITY

7.6 How to go to create your own project in Help.

EXERCISE 7.6 • LEVEL: BASIC • USE HELP TO CREATE YOUR OWN PROJECT

- Click the Help menu.
- Choose Getting Started.
- Click on "Project Map."
- Click on the "Build a Plan" directory.
- Click on "Define a Project" subdirectory.
- Click on "Initiate a project" file

- Work through the other directories in the same way applying the instructions to your own project.

Dialog

Explain that create your own project is an excellent way to find out how to build a good project plan in MS Project. It also provides answers to specific questions as well as providing more advanced exercises on managing projects.

A nicely
structured
tutorial
that covers
the basics
of using
MS Project

Tutorial

2000 is accessed from the Help menu's Getting Started, Quick Preview.

This is the context sensitive way of getting into the help system. At any point

Shift F1

in the application you can hit the F1 function key and context sensitive Help information will be displayed. To get further information concerning a warning

Errors/warnings or error message,
press the
F1 key
while the
message is
displayed.

MODULE 8: USING VIEWS

Overview

The using views module introduces how to manipulate the way data is being displayed to suit the work being done. This module discusses how views are used by MS Project 2000 to provide control and flexibility. It also reviews the use of tables and forms.

Objectives

The participant will:

- Be able to split the screen.
- Be able to access the underlying database.
- Be able to add customized fields to a table.
- Understand how information is displayed in Microsoft Project 2000.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise. The level is suitable for basic users.



ACTIVITY

8.1 How to change to a new view.

EXERCISE 8.1 • **LEVEL: BASIC** • **CHANGE TO A DIFFERENT VIEW**

- Pick View menu.
- Choose Calendar.
- Pick View menu.
- Choose Network Diagram.
- Pick View menu.
- Choose Gantt chart.

Dialog

Explain that the variety of views only

change the way the user sees the information in the underlying database. Note that one can work in any view and that each view can be printed exactly as it is seen on the computer display screen.

Up to two views can be displayed at one time and when information is changed in one view, it is reflected in the other view showing on the computer display screen. Views are a most powerful feature of MS Project 2000. The view is the means of entering, changing and displaying the contents of the underlying database in MS Project 2000. By using a variety of views, you can look at the same project information in different ways as you organize your project. There are two types of view: single pane and combination.

In a combination view, MS Project

2000 binds together the data in the two different parts of the combination. When you move from one row to another in the top view (using either the arrow keys or selecting with the mouse), the data displayed in the bottom view will reflect the change.

You use the View menu to change from one view to another.

Alternatively, you can click on the buttons on the left of your screen to go from one view to another.

ACTIVITY

8.2 How to split the computer display screen to show two views at a time.

8.3 How to return to a single view by removing the split.

8.4 How to split the computer display screen without using the menu bar.

EXERCISE 8.2 • LEVEL: BASIC • CHANGE TO A COMBINATION VIEW, METHOD 1

- Pick Window menu.
- Choose Split.

EXERCISE 8.3 • LEVEL: BASIC • CHANGE TO A SINGLE VIEW

- Pick Window menu.
- Choose Remove Split.

EXERCISE 8.4 • LEVEL: BASIC • CHANGE TO A COMBINATION VIEW, METHOD 2

- Move the cursor to the inside square box in the bottom right corner. (The shape of the cursor will change to a double arrow).
- When the cursor changes, hold down the mouse button and drag up.

Dialog

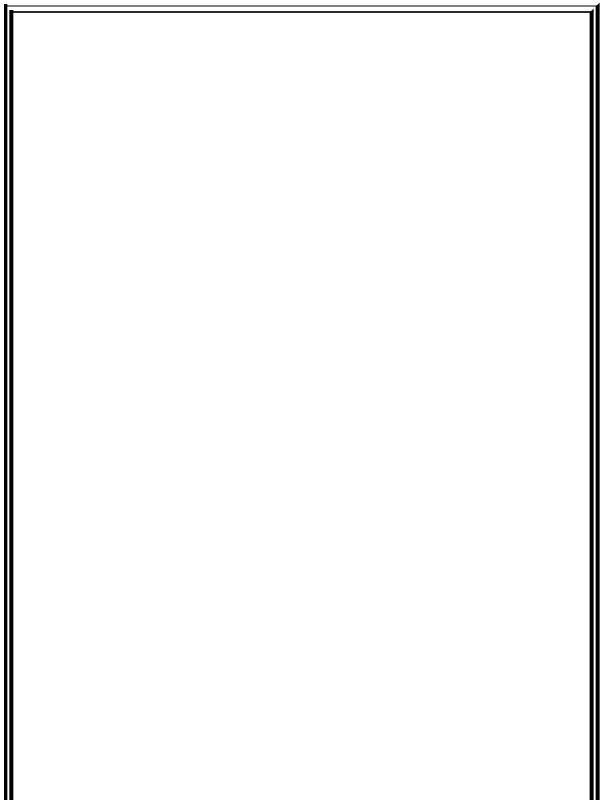
Explain that when you display two views at a time, you can determine which view you are working in by the

dark blue border on the left-hand side (LHS) of the screen. There are a couple of ways to split and unsplit your computer display screen.

You change to a combination view by using the Window menu. Note that only one view of a combination view is current at a

Combination

time. This is indicated by the blue border down the LHS. You can also change to a combination view by dragging in the small square in the bottom right-hand corner of the window.



ACTIVITY

8.5 How to alter the single view to see more or less of the table section of the Gantt chart.

EXERCISE 8.5 • **LEVEL: BASIC • LOOKING AT THE TABLE BEHIND THE GANTT CHART**

- Move the cursor to the vertical line at the LHS of the Gantt chart. The shape of the cursor will change to a double arrow.
- When the cursor changes, hold down the mouse button and drag right.

Dialog

Explain that when the cursor changes shape, it allows you to do an number

of different things, such as show more or less of the table section of the Gantt chart. There are a number of predefined tables already set up in MS Project; however, each can be altered to suit the requirements of the user.

Manipulating views

You change from a combination to a single view by dragging the horizontal divider line to the bottom of the screen. Tables are

Tables

spreadsheets used in views to display information on either tasks or resources. The rows are either task or resource names and the columns are fields of information about the task or

resource. We
will return to
tables in a
later
exercise
(see [Module
12](#)).

ACTIVITY

8.6 How to add another view to the View menu list.

EXERCISE 8.6 • LEVEL: BASIC • PUTTING THE TASK SHEET VIEW ON THE VIEW MENU

- Pick View menu.
- Choose More Views.
- Select Task Sheet.
- Click Edit button.
- Select Show in Menu option.
Click OK button.
- Click Close button.

- Pick View menu.
- Choose Task Sheet.

Dialog

Explain that the views that are used most often by the user can be added to the View menu list. This means that views can also be removed, as well.

MODULE 9: SETTING TASK DEPENDENCIES

Overview

This module introduces how to set task linkages or dependencies (called relationships in MS Project 2000) using a number of different methods. The user will also look at the types of relationships that exist and learn how to use best practice in setting task dependencies.

Objectives

The participant will:

- Be able to link tasks.
- Be able to set the critical path and format text to highlight critical path information.

Preparation

Review the contents of this module and do each of the

exercises at least once to familiarize yourself with the logic and outcome.

Dialog

Explain that tasks can be linked in a number of ways and in a number of views. MS Project automatically sets relationships as one finishing before another one begins. This is the most common type of relationship between tasks, as one can see from the example of building a house the foundation must be finished before the walls can be built.

Task relationships

The most common relationship between tasks is when one task cannot start until another task finishes. An obvious example of this type of relationship is during the building of a house when the foundations must be dug

before the concrete can be poured, and the concrete must harden before the walls can be built. MS Project 2000 assumes we want this type of relationship unless we tell it otherwise.

There are two basic ways of setting

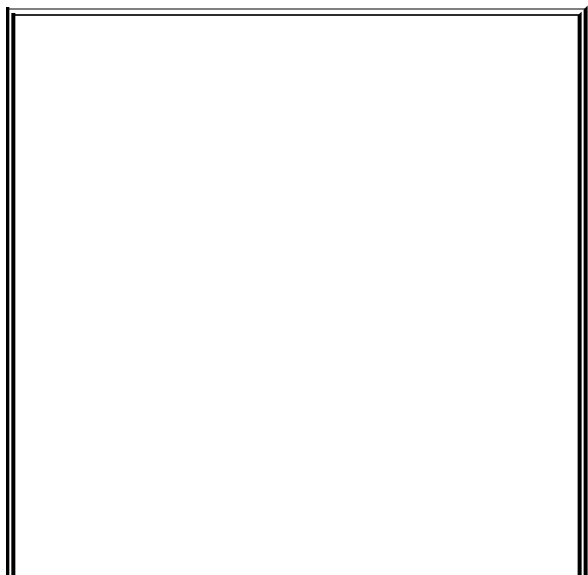
Setting relationships relationships in

MS Project 2000:

1. Select the tasks in a table and tell MS Project to link them.
2. Use the Network Diagram view and use the cursor to connect boxes.

Presentation

The activities in this module cite the aims of each exercise, and whether the level is suitable for basic or advanced users.



ACTIVITY

9.1 How to highlight the critical path for a MS Project file.

9.2 How to highlight the text associated with the critical path in an MS Project file.

EXERCISE 9.1 • **LEVEL: BASIC** • **CHANGE GANTT CHART TO HIGHLIGHT CRITICAL PATH**

- Open **Golf1C**.
- Click Gantt chart wizard icon.
- Choose Critical Path as information to be displayed.
- Click Finish button and click Format It button.

- Exit Wizard.

EXERCISE 9.2 • **LEVEL: BASIC** • **AMEND TEXT STYLE**

- Pick Format menu.
- Choose Text Styles.
- Item to change = Critical Tasks.
- Color = red.
- Click OK button.

Dialog

Explain that it is useful to have the critical path information showing in a project plan. MS Project can highlight this information automatically, and

will change the tasks that have been highlighted as the critical path changes in a project. Before we start setting task dependencies, we will first amend the format of the display to show more clearly the critical path.

Gantt chart wizard

This is an interactive assistant to help us format the Gantt chart view to the way we want it.

Note that some of the Gantt lines are in blue and some are in

Critical path red. The tasks with red lines on the Gantt chart are those tasks which are on the critical path.

ACTIVITY

9.3 How to set dependencies between tasks using the menu bar.

EXERCISE 9.3 • **LEVEL: BASIC** • **USING GANTT VIEW AND MENU**

- Use the cursor to select Tasks 1, 2 and 3.
- Pick Edit menu.
- Choose Link Tasks.

ACTIVITY

9.4 How to set dependencies between tasks using the link tasks icon.

EXERCISE 9.4 • **LEVEL: BASIC • USING GANTT CHART VIEW AND TOOLBAR**

- Use the cursor to select Tasks 4, 5 and 6.
- Click on the link tasks icon on the toolbar.

ACTIVITY

9.5 How to set dependencies between tasks using the bars in the Gantt chart.

EXERCISE 9.5 • **LEVEL: BASIC** • **USING GANTT CHART VIEW**

- Click and hold the center of the task bar for Task 6.
- Drag directly down to the center of the task bar for Task 7.

ACTIVITY

9.6 How to show more of the overall schedule or to show more of the detail.

EXERCISE 9.6 • **LEVEL: BASIC** • **USING THE ZOOM ICONS**

- Click on the zoom out icon on the toolbar to see more of the schedule.

Dialog

Explain that the zoom in and out icons are one way to manipulate the image one sees on the computer display screen. There is also a Zoom command under the View menu.

ACTIVITY

9.7 How to go to the current finish date of an MS Project file.

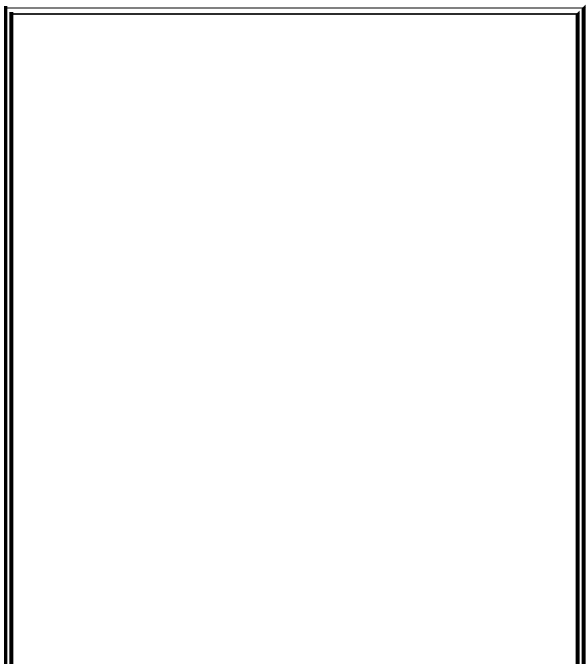
EXERCISE 9.6 • **LEVEL: BASIC** • **FINDING THE PROJECT END DATE**

- Pick Project menu.
- Choose Project Information command.
- Click Statistics button.
- Look at and note current finish date.

Dialog

Explain that the project end date will change as the project changes and

one way to find the end date of the project is to use the statistics button in the project information box.



ACTIVITY

9.8 How to set dependencies by using the predecessor column in the task sheet.

EXERCISE 9.8 • LEVEL: ADVANCED • USING THE VIEWS, TASK SHEET AND NETWORK DIAGRAM

- Pick Window menu.
- Choose Split to get combination view.
- Select bottom view by clicking on it.
- Pick View menu.
- Choose More views.

- Select Network Diagram. Apply.
- Select top view by clicking on it.
- Pick View menu.
- Choose More views.
- Select Task Sheet. Apply.
- Select the predecessor column of Task 4.
- Enter 3 in the predecessor column and hit CR.
- What is the current finish date of the project?

ACTIVITY

9.9 How to set dependencies by using the ID column in the Task Form.

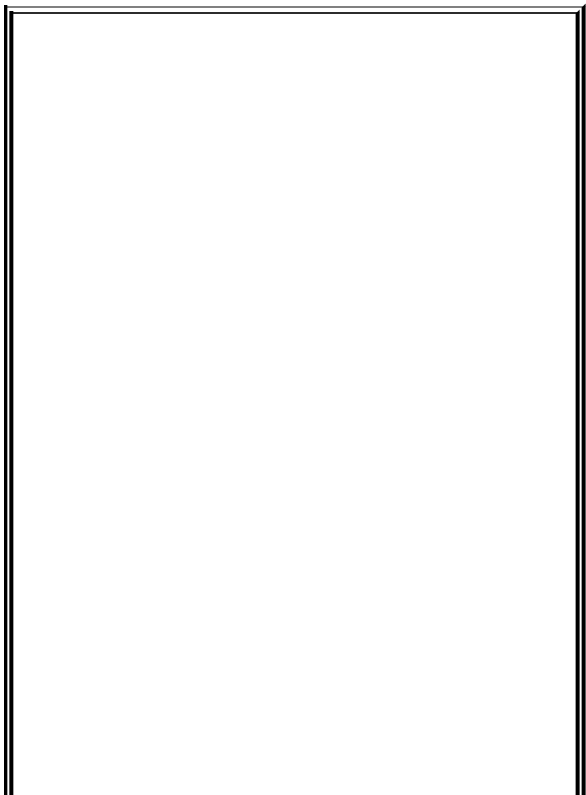
EXERCISE 9.9 • LEVEL: ADVANCED • USING THE VIEWS, GANTT CHART AND TASK FORM

- Select the bottom view (Network Diagram).
- Pick View menu.
- Choose More views.
- Select Task Form. Apply.
- Pick Format menu.
- Choose Details.

- Pick predecessors and successors.
- Click Previous or Next buttons on bottom view to select Task 6.
- On bottom view, enter 7 in ID column of successor.
- Click OK.
- What is the current finish date of the project?

Dialog

Explain that the user can set relationships between tasks by entering the ID number of the predecessor of the task in the appropriate column.



ACTIVITY

9.10 How to set dependencies using the network diagram.

Comment

Still using the file Golf1C.

EXERCISE 9.10 • **LEVEL: BASIC** • **USING THE VIEW, NETWORK DIAGRAM**

- Pick Windows menu.
- Choose remove split.
- Pick View menu.
- Choose Network Diagram view.
- Click in the box for Task 7 and

drag the cursor to the box for Task 8.

- What is the current finish date of the project?

ACTIVITY

9.11 How to set dependencies between tasks using Gantt chart view.

Comment

Note still using the file Golf1C.

EXERCISE 9.11 • **LEVEL: BASIC** • **USING THE VIEW, GANTT CHART**

- Pick View menu.
- Choose Gantt chart view.
- Select Task 8.
- Click on go to select task icon (this will cause the the Gantt chart to scroll such that the bar

for Task 8 appears on the screen).

- Click on Gantt bar of Task 8 and drag the cursor to the bar of Task 9.
- What is the current finish date of the project?

ACTIVITY

9.12 Practice exercise in linking tasks.

EXERCISE 9.12 • **LEVEL: BASIC** • **SET RELATIONSHIPS**

- Enter relationships for the remaining tasks using whichever method you prefer.
- Which are the non-critical tasks?

There are
as many
ways of
canceling a
Canceling a relationship

relationship as there
are of
setting it
up.

ACTIVITY

9.12 How to remove task dependencies using the menu bar.

9.13 How to remove task dependencies using the unlink tasks icon.

9.14 How to remove task dependencies in the PERT chart view.

EXERCISE 9.12 • **LEVEL: BASIC** • **USING THE GANTT VIEW AND THE MENU**

- Use the cursor to select Tasks 1, 2 and 3.
- Pick Edit menu.
- Choose Unlink Tasks.

EXERCISE 9.13 • LEVEL: BASIC • USING THE GANTT VIEW AND THE TOOLBAR

- Use the cursor to select Tasks 4, 5 and 6.
- Click on the unlink tasks icon on the toolbar.

EXERCISE 9.14 • LEVEL: BASIC • USING THE PERT CHART

- Pick View menu.
- Choose Network Chart view.
- Double click on the line joining Tasks 7 and 8.
- Choose Delete.

Dialog

Explain that there are a number of different ways to remove the relationships between tasks.

ACTIVITY

9.15 How to set dependencies for all the tasks in an MS Project file.

EXERCISE 9.15 • **LEVEL: BASIC** • **LINK ALL TASKS IN FINISH-TO-START RELATIONSHIP**

- Pick View menu.
- Choose Gantt Chart view.
- Select all tasks by clicking on the header of the Task Name column.
- Click on unlink tasks icon.
- Click on link tasks icon.

Dialog

Explain that one can link or unlink all the tasks in an MS Project file by merely highlighting the Task Name column and choosing the appropriate icon or command.

MODULE 10: USING ORGANIZATON AND PROJECT CALENDARS

Overview

The using organization and project calendars module introduces how calendars are used by Microsoft Project 2000. The main lesson in this module is to tailor a calendar and associate it with the current project.

Objectives

The participant will:

- Know how to amend a standard calendar.
- Know how to set a standard calendar as the project calendar.

Preparation

Review the contents of this module and do each of the

exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims for each exercise. The level is suitable for basic users.



ACTIVITY

10.1 How to add public holidays to the standard calendar.

EXERCISE 10.1 • **LEVEL: BASIC** • **UPDATE STANDARD CALENDAR TO INCLUDE PUBLIC HOLIDAYS**

- Pick Tools menu.
- Choose Change Working Time command.
- Click on scroll arrow to move to October.
- Click on last Monday.
- Select Non-working Time.
- Repeat for all Public Holidays for

the year ahead.

New Year's Day

January 15

February 19

May 28

July 4

September 3

October 8

November 12

November 22

Christmas Day

Dialog

Explain that MS Project is set up for a 40-hour working week with no work taking place on Saturdays or Sundays. No public holidays have been entered into MS Project.

Calendars are used by the scheduler to map task and

Calendars project durations onto an actual set of dates. The calendar allows for

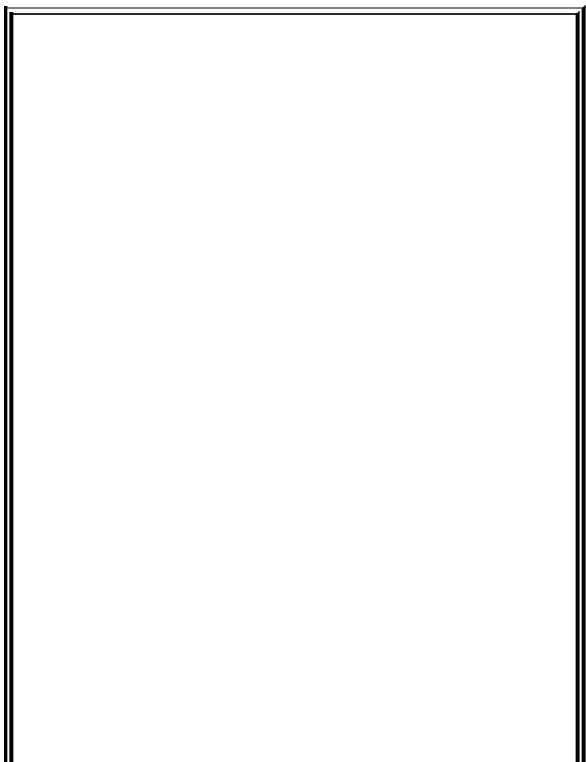
- public holidays
- company holidays
- weekends
- and working hours

to be accounted for by the scheduler.

MS Project 2000

provides a
Standard
Calendar which
can then be
amended as
required by the
user. It is
recommended
that you have
one person
within an
organization
amend the
standard
calendar and
provide it to all
project

managers.



ACTIVITY

10.2 How to create a new project calendar using the standard calendar.

EXERCISE 10.2 • **LEVEL: BASIC** • **AMEND PROJECT CALENDAR**

- Pick Tools menu.
- Choose Change Working Time.
- Select Standard (Project Calendar).
- Click on New button.
- Name the new calendar "Golf."
- Select "M to Th" on the heading line. This selects every Monday, Tuesday, Wednesday and

Thursday in the calendar.

- Change finishing hours to "19:00."
- Click on F and change to Non-working Time.
- Click on OK button.

Dialog

Explain that project calendars can be used to reflect information specifically related to a project such as the working hours or non-working days. The workhours are used by the scheduler to understand how many hours of work are available in each day to complete the amount of work which has been calculated for each task. For this project we will work a 4-day week, 10-hour day. We set

workhours for all Mondays through Thursdays to finish at 19:00 and set Fridays as non-working.

ACTIVITY

10.3 How to ensure that the project is using the appropriate calendar.

EXERCISE 10.3 • **LEVEL: BASIC** • **RESET PROJECT CALENDAR**

- Pick Project menu.
- Choose Project Information command.
- Calendar = "Golf."
- Click OK button.

Dialog

Explain that if you do not reset the project calendar, the project will always use the standard calendar as

the default. Having created a new calendar we must now have the project use this calendar. What is the current finish date of the project? We are still working 40 hours per week so why has this changed?

MODULE 11: OUTLINING TASKS

Overview

The outlining tasks module introduces the concept of outlining tasks so that a project can be viewed at a micro and macro level of detail. This module also looks at how to insert a recurring task.

Objectives

The participant will:

- Be able to structure a project to include summary levels.
- Be able to view a project at either a summary level or a subtask (detail task) level.

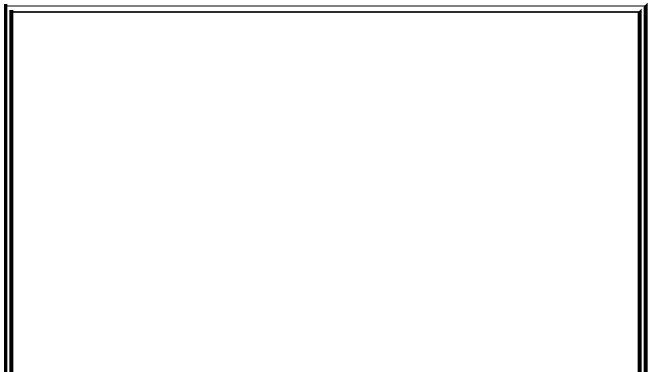
Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the

logic and outcome.

Presentation

The activities in this module cite the aims for each exercise. The level is suitable for the basic users.



ACTIVITY

11.1 How to insert summary or outline tasks.

EXERCISE 11.1 • LEVEL: BASIC • INSERTING SUMMARY TASKS

- Open **Golf1D**.
- Select "Clear land."
- Pick Insert menu.
- Choose new task.
- Type in "Phase 1 Earthmoving."
- Select "Clear land."
- Click indent button on formatting

toolbar.

- Note change in duration of Phase 1.
- Note change in bar of Phase 1.
- Select Tasks 5, 6 and 7 and click indent button.
- Repeat for Phase 2, before "Dig trenches."
- Repeat for Phase 3, before "Final grading."
- Remove final milestone from Phase 3 by outdenting the task.

Dialog

Explain that summary or outline tasks serve to divide the project in sections

that can be used to report higher-level information to management. The duration of the Summary task is never specified because the durations for all the detailed tasks beneath it are rolled up into the duration figure in the summary task.

We have reached the stage in our planning when we need to examine the overall plan

Summary tasks and report to our manager on it. We currently have just a list of tasks so the first step is to organize these a little better.

We will identify 3 phases in our project

- Phase 1:
Earthmoving
- Phase 2:
Drainage/Irrigation
- Phase 3:
Landscaping

We will insert these as outline tasks (another name for summary tasks) in our project. A summary task does not exist as a

specific task. It is a way of summarizing information about the detailed tasks indented below it.

--

ACTIVITY

11.2 How to hide or show detailed tasks.

EXERCISE 11.2 • **LEVEL: BASIC** • **HIDE AND SHOW SUBTASKS**

- Select all tasks.
- Click on hide subtasks icon on formatting toolbar ().
- Select Phase 1 task.
- Click on show subtasks icon on formatting toolbar ().
- To show all, click on show all subtasks icon on formatting toolbar ().

- To hide or show subtasks, click on + next to Summary task name.

Dialog

Explain that the user can manipulate the information seen on the computer display screen by rolling up some or all of the subtasks or detailed tasks within the summary tasks. This is done by using the small boxes next to the summary task name which have either a - or a + sign in them. Having created some summary tasks we can now create some useful displays of our project tasks.

ACTIVITY

11.3 How to enter a recurring task.

EXERCISE 11.3 • LEVEL: BASIC • ENTER A RECURRING TASK

- Select "Clear land."
- Pick Insert menu.
- Choose Recurring task command.
- Name = "Status Meeting."
- Duration = "2 hours."
- This Occurs = "Monthly, 1st Monday."

- Insert Start date and note the number of occurrences as 6.
- Click OK button.
- Note the warning message that appears and choose the appropriate response.

Dialog

Explain that tasks such as meetings or the preparation of status reports that happen at the same time each week or month can be inserted into the plan by using the Recurring Task command. These tasks can be altered or tracked the same way as normal tasks.

We

Recurring task

sometimes
want to
schedule a
task to occur
on a regular
basis, for
example, a
status
meeting. We
can easily do
this with MS
Project
2000.

ACTIVITY

11.4 How to show a summary of the entire project.

EXERCISE 11.4 • **LEVEL: BASIC • DISPLAY SUMMARY AT PROJECT LEVEL**

- Pick Tools menu.
- Choose Options command.
- Select View tab.
- Select box = Project Summary.
- Click OK button.

Dialog

Explain that instead of indenting all

the summary tasks under a summary task that is to represent the entire project, MS Project can do this for you automatically. The project summary can be copied and pasted to another file with other projects on it.

ACTIVITY

11.5 How to show numbers like a table of contents on the MS Project plan.

EXERCISE 11.5 • **LEVEL: BASIC** • **DISPLAY OUTLINE NUMBERING**

- Pick Tools menu.
- Choose Options command.
- Select View tab.
- Select box = Show Outline Number.
- Click OK button.

Do you recognize the Work Breakdown Structure?

Dialog

Explain that sometimes it is convenient to show a numbering system for your task list. This can be inserted automatically by MS Project 2000.

MODULE 12: PRINTING VIEWS

Overview

The printing views module introduces how to format the Gantt chart in order to include specific information to be printed on it. This module also looks at creating a new table that could be printed by the user.

Objectives

The participant will:

- Be able to set up the information to be printed on the Gantt chart.
- Be able to create a new table.

Preparation

Review the contents of this module and do each of the

exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims for each exercise and whether the level is suitable for basic or advanced users.



ACTIVITY

12.1 How to alter the width of a column.

EXERCISE 12.1 • **LEVEL: BASIC** • **CHANGING COLUMN WIDTH**

Method 1

- Click on dividing line on right-hand side (RHS) of Task Name column and drag.

Method 2

- Double click on dividing line on RHS of Task Name column.
- The width will automatically adjust to best fit.

Dialog

Explain that it is convenient to be able to change the width of a column either when the text has been cut off or when the column is too narrow for the numbers within it and #### appear in the cell.

ACTIVITY

12.2 How to alter the computer display screen to set what will be printed.

EXERCISE 12.2 • **LEVEL: BASIC** • **PRINT GANTT CHART VIEW**

- Amend the layout on screen to show up to Duration column.
- Pick File menu.
- Choose Print command.
- Click on Print Preview button.
- Note the cursor changes to a magnifying glass.
- Click the mouse to examine the

output in more detail.

Dialog

Explain that when the project file is printed it will only show the information that you have specified. If you drag the calendar section of the Gantt chart off to the RHS of your screen only the Gantt chart table will print.

We now want to print the project plan in order to let others examine it. We will print this information in

Printing two different ways. First, we will print the Gantt chart and second, we will print a list of the tasks with their start and finish dates.

MS Project 2000 provides two printing features, print views and print reports. We will use

the print views in this section. As the name implies, this prints the view currently displayed on the screen.

For each type of view and report we can define the header, footer, margins, fonts, colors, etc. We will use the defaults

Page setup

initially.

The preview functionality allows us to view on the screen the appearance of the report prior to us sending it to the printer.

Note that you should *always* preview before you print in order to check how many

**Print
preview**

pages you will have. Note also that the Print functionality allow you to print information between certain dates.

For this Gantt chart we will print a *single* page giving the project overview. We will show 3

columns of
data: the task
ID, the task
name and the
duration. We
will not print a
legend.

ACTIVITY

12.3 How to alter the timescale used when viewing/printing a Gantt chart.

EXERCISE 12.3 • **LEVEL: BASIC** • **PRINT ANOTHER GANTT CHART VIEW**

- Amend the layout on screen to show up to Duration column.
- Pick Format menu. Choose Timescale command.
- Under Major Scale, pick Units: = days, Count 1.
- Under Minor Scale, pick Units: = hours, Count 6.
- Click OK button.

- Pick File menu.
- Choose Print command.
- Click on Print Preview button.

(We are showing the legend which we don't want)

- Click the Page Setup button
- Click the Legend Tab
- Legend on = None, Click the OK button
- Click the single page and multiple page icon to see the extent of the print.

Dialog

Explain that sometimes the user wishes to see more or less detail when they print the Gantt chart. Changing the timescale is a means to reflect the level of detail required.

For this Gantt chart we will print multiple pages showing the project with a daily timescale. We will show three columns of data: the task ID, the task name and the duration. We will not print a legend.

ACTIVITY

12.4 How to create a new table in the MS Project file.

EXERCISE 12.4 • **LEVEL:**

ADVANCED • CREATE NEW TABLE

- Pick View menu.
- Choose Table command.
- Pick More tables. Select Entry.
- Click copy button and change name to Task & List.
- Select the Title column of Name Field Name entry.
- Change *Task Name* to *Activity Name*.

- Select the Width column of Predecessors Field Name entry and change to 8.
- Select the Title column of Predecessors Field Name entry and set to *Pred*.
- Select the Title column of Duration Field Name entry and set to *Effort*.
- Select Name Field Name entry. Select insert row button.
- Select the option button (the drop-down arrow at the right-hand end of the blank edit line).
- Scroll down to WBS (Work Breakdown Structure) and select.

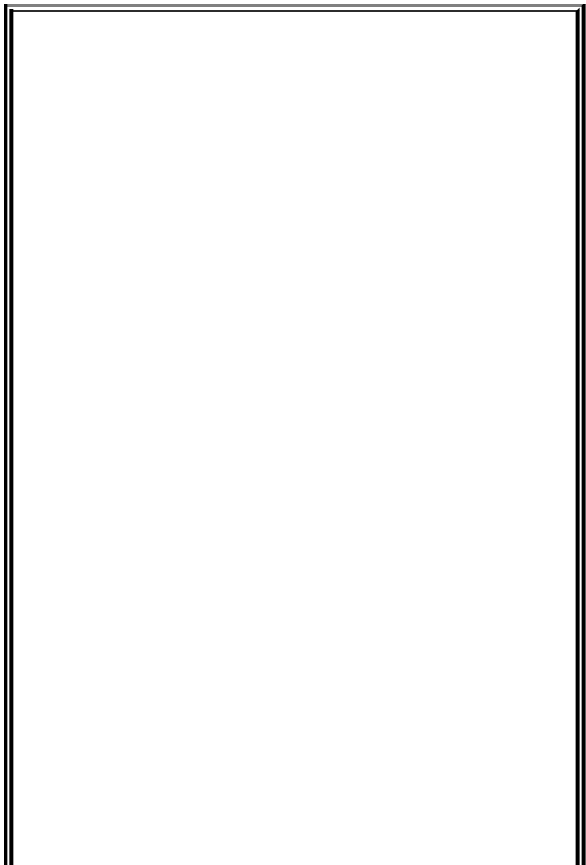
- Select Resource Names Field Name entry.
- Click delete row button. Click OK button.

Dialog

Explain that the user may wish to create a table that shows specific information from the database.

Tables can be created or revised to suit the user.

Before we print the table of dates we need to amend the layout. We will create a new table to suit our own requirements. The information we need is similar to that in the task sheet table but we don't need the resource details.



ACTIVITY

12.5 How to look at the new table in the view display.

EXERCISE 12.5 • **LEVEL: ADVANCED • CREATE A VIEW DISPLAYING NEW TABLE**

- Pick View menu.
- Choose More Views command.
- Select Task Sheet. Apply.
- Pick View menu.
- Choose Table command.
- Select More Tables.

- Select Task List. Apply.

Dialog

This exercise shows the user how to see the new table.

ACTIVITY

12.6 How to format information that is to be printed.

EXERCISE 12.6 • **LEVEL:**

**ADVANCED • FORMAT AND PRINT
NEW TABLE**

- Pick File menu.
- Choose Page Setup command.
- Click on Header tab, click on Font button (A).
- Item to change = Header Center (Line 1).
- Font = Times New Roman, Size = 14, Bold.

- Item to change = Header Center (Line 2).
- Font = Times New Roman, Size = 10, Italics.
- Click OK button.

Insert a second Header Line

- Move cursor after & *[Project Title]*, hit CR.
- Enter *Task List*, hit CR.
- Add line for project manager.
- Pick Print Preview and view the changes.

Insert a Footer Line

- Click on Footer tab.
- Choose Left tab.
- Type *Printed on:* and, using the icons, add date, time.
- Choose Right tab.
- Go to drop-down list and pick *File name* =. Add.
- Pick Print Preview and view the changes.
- If you are satisfied with the result, pick Print.

Dialog

This exercise shows how to format information that is to be printed. The

formatting details will be saved in the MS Project file memory.

MODULE 13: ASSIGNING RESOURCES

Overview

The assigning resources module introduces how to assign one or more resources to tasks. It also looks at how costs are assigned to resources and how to amend resource calendars.

Objectives

The participant will:

- Be able to assign resources to tasks.
- Be able to identify overallocations.
- Know how to amend the work units of a resource.
- Be able to assign costs to a resource.
- Be able to amend a resource calendar.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Resource

Resources are the people, equipment and supplies used to complete tasks in a project. A resource is considered by MS

Project 2000 to be a resource type and there can be one or many units of each type.

This is the allocation of a resource type to work on a particular task.

Assignment MS Project 2000 assigns one unit (or 100 percent) of a resource to work full time for the duration of a

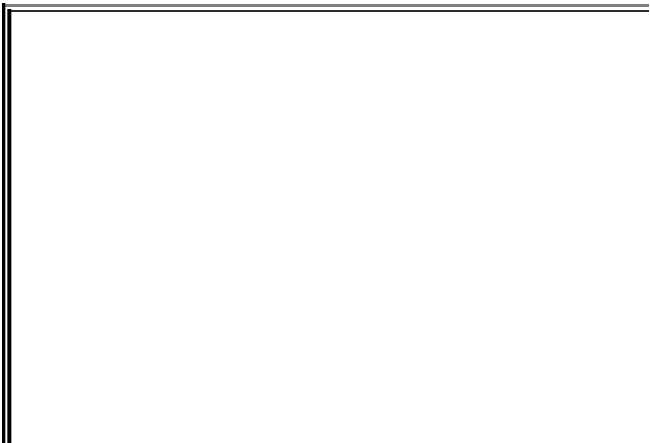
task.

Creating

You can create resources as a separate exercise or you can create them as you are assigning them to a task. You can create a single resource (with max units =1 or 100 percent) or a group of resources (with max. units > 1 or greater than 100 percent).

Presentation

The activities in this module cite the aims of each exercise and whether the level is suitable for basic or advanced users.



ACTIVITY

13.1 How to assign a resource using the menu bar.

13.2 How to assign a resource using the task information box.

13.3 How to assign a resource using assign resources icon on the standard toolbar.

13.4 How to assign a resource using the task form view.

13.5 How to assign more than one resource using the assign resources icon.

EXERCISE 13.1 • **LEVEL: BASIC** • **ASSIGN A SINGLE RESOURCE,** **METHOD 1**

- Open **Golf1E**.

- Select Task 4, "Clear land."
- Pick Tools menu.
- Choose Resources command, Assign Resources.
- An Assign Resources box will appear.
- In the first cell, type "Bulldozer." Click CR.
- Select "Bulldozer." Click Assign.
- Click Close.
- The "Bulldozer" is now assigned to Task 4.

**EXERCISE 13.2 • LEVEL: BASIC •
ASSIGN A SINGLE RESOURCE,**

METHOD 2

- Select Task 5, "Rough grading".
- Double click on Task 5 to produce the task information box.
- Choose the Resources tab.
- Select the first cell and type in "Bulldozer driver."
- Click OK.
- The Bulldozer driver is now assigned to Task 5.

**EXERCISE 13.3 • LEVEL: BASIC •
ASSIGN A SINGLE RESOURCE,
METHOD 3**

- Select Task 6, "Dig lakes and water hazards."
- Choose the assign resources icon from the standard toolbar.
- Add "JCB."
- Click Assign.
- Click Close.
- The JCB is now assigned to Task 6.

**EXERCISE 13.4 • LEVEL: BASIC •
ASSIGN A SINGLE RESOURCE,
METHOD 4**

- Select Task 7, "Dig bunkers."
- Pick Window menu.

- Choose Split command.
- Select the space directly under resource name.
- A drop-down arrow appears.
- Click on the drop-down arrow and choose "JCB."
- Click OK.
- The JCB is now assigned to Task 7.

EXERCISE 13.5 • **LEVEL: BASIC** • **ASSIGN MULTIPLE RESOURCES**

- Select Task 9, "Dig trenches."
- Choose the assign resources icon from the standard toolbar.

- Add "JCB driver" into the next blank cell.
- Select "JCB" and "JCB driver."
- Click Assign.
- Click Close.
- Both the JCB and the JCB driver are now assigned to Task 9.

Dialog

Explain that there are a variety of ways to assign a resource to a task. A resource could be a piece of equipment as well as being a person. Note that MS Project 2000 automatically assigns the resource full-time.

Tasks that are scheduled at the same time for the same resource means that the resource is scheduled for more hours in the day than we have specified in the workhours

Overallocation section of the calendar. MS Project 2000

has many ways to display overallocation. An overallocated resource will be highlighted in red in any one of the resource views.

ACTIVITY

13.6 How to identify an overallocated resource.

EXERCISE 13.6 • **LEVEL: BASIC** • **VIEW RESOURCE USAGE**

- Open **Golf1G**.
- Pick View menu.
- Choose Resource Sheet command.
- Note that certain resources appear in red.
- Pick View menu.
- Choose Resource Graph.

- Use the horizontal scroll bar on the left to scroll through the resources until you reach the JCB which will be in red.
- Use the horizontal scroll bar on the right to scroll through to the week of June 26.
- Note the amount of the overallocation.
- Pick the Format menu.
- Choose Details, Work.
- The vertical axis should change to the days that the resource has been allocated, showing the overallocation in red.

Explain that when a resource is overallocated it appears in red on the resource sheet, the resource graph and the resource usage views. The resource graph shows the number of units the resource is overallocated by, but the user can change this to a variety of other views using the Details command.

ACTIVITY

13.7 How to amend the work units of a resource.

EXERCISE 13.7 • **LEVEL: BASIC** • **AMENDING WORK UNITS**

- Pick File menu. Choose New.
- Create Tasks 1, 2 and 3, each with a duration of 2 weeks. Click CR.
- Select Task 1.
- Click the assign resources icon.
- Enter "Joe" in the first cell and click Assign.
- Select Task 2.

- Note that the assign resources box remains open.
- Select "Joe" and enter "5 units" beside his name.
- Click Assign or click CR.
- Close assign resources box.
- Select Task 3.
- Pick Window menu. Choose Split.
- In the Task Form View on the bottom part of the screen, add "Fred, .5 units" under Resource Name.
- Click OK.

- Note the hours of work associated with Tasks 1, 2 and 3.
- Select Task 1.
- Change the units to 50 percent. Note the change in the duration of the task.
- Select Task 2.
- Change the units to 100 percent. Note the change in the duration of the task.

Dialog

Explain that MS Project uses a formula to calculate the amount of time it will take to complete a task. This amount of time is called work,

and can only be determined when the units per resource is multiplied by the duration of that task.

$$\text{units} * \text{duration} = \text{work}$$

Units indicate the availability of a resource. When a user builds a plan, each task is assigned a duration which estimates how much time it will take to complete the task. If a resource is available only 50 percent of the time, however, the task will take twice as long to complete.

Amend the units per resource when you know the availability of that resource to see what effect it has on the project plan.

The main source
of expense in
projects tends to

Costs

be the cost of the resources. MS Project 2000 automatically calculates this cost when calculating work.

ACTIVITY

13.8 How to assign costs to a resource.

EXERCISE 13.8 • LEVEL: BASIC • ASSIGN COSTS TO RESOURCES

- Open **Golf1F**.
- Pick View menu.
- Choose Resource Sheet.
- In Standard Rate column, enter hourly or daily rates for these resources:

Bulldozer	300/d
Bulldozer	

driver	30/h
--------	------

JCB	250/d
-----	-------

JCB driver	25/h
------------	------

Crew	6/h
------	-----

- Pick Project menu. Choose Project Information command.
- Click Statistics button.
- Note information in cost column.
- Click on Close button.
- Pick View menu.
- Choose Table command.

- Select Cost.

Dialog

Explain that costs can be assigned per resource, so that this information can be used to estimate the total costs of a project and to track the costs of a project as it progresses.

ACTIVITY

13.9 How to amend the calendar to show time off for one resource.

EXERCISE 13.9 • **LEVEL: BASIC** • **AMENDING RESOURCE CALENDARS**

- Open **Golf1 F.**
- Select "Clear land" and "Rough grading" tasks.
- Note end dates of tasks.
- Pick Tools menu. Choose Change Working Time.
- For: = Pick bulldozer.
- Set 3 days non-working in May

for maintenance and click OK.

- Note change in end dates for the two tasks.

Dialog

Explain that once resources are assigned to a project, a calendar for each is automatically created by MS Project. This is used for specifying annual leave and for defining work hours for each resource.

This is specific to a particular resource type. It can be used for maintaining holiday information and also for defining working hours. For

Resource each resource type

calendar we can specify a different set of working days and/or hours. MS Project 2000 will then take these into account when scheduling tasks.

ACTIVITY

13.10 How to manipulate multiple resources Method 1.

13.11 How to manipulate multiple resources Method 2.

EXERCISE 13.10 • **LEVEL: ADVANCED • MULTIPLE RESOURCES NO. 1**

- Pick File menu.
- Choose New command.
- Pick View menu.
- Choose Resource Sheet.
- Add entries for "JCB, Bulldozer, Operator," each with max units

= "5 units" (500 percent).

- Pick View menu. Choose Gantt chart.
- Pick Window menu. Choose Split.
- Go to Format menu. Pick Timescale. Display timescale in days.
- Create Task 1, 2 weeks. Click CR.
- Select Task 1. Click on resource assignment icon to display dialog box.
- Choose "JCB" from name column and click on assign button. Close box.

Note that the JCB has 80 hours of work to accomplish in 2 weeks duration.

- We decide this is too long, we need it completed sooner, so we will use two JCBs.
- In Task Form, change JCB availability to 2 units (200 percent). Click OK.
- The new duration is 1 week.

Note that JCB work hours of 80 are driving this duration.

- Create Task 2, 2 weeks.
- Select Task 2. Pick Tools menu, Resources and select Assign Resources.

- Add "Bulldozer, 1 unit" (100 percent) and "Operator, 1 unit" (100 percent).
- Close Box.

Note that Operator and Bulldozer are driving the duration of 2 weeks.

- In Task Form (bottom view), change Bulldozer to 2 units (200 percent).
- Note that Operator is now the driving resource.
- To see this clearly, apply the resource usage view to the bottom view.

To get back to default view, pick Views, More Views, and Task Form.

Click Apply.

- Change Operator to 2 units (200 percent). Click OK.
- Duration is 1 week as Operator and Bulldozer are both driving resources.
- Still on Task 2, add "Foreman, 0.5 units" (50 percent), to Task Form (bottom view).
- Click OK button.

Note hours. Note that Operator and Bulldozer are still driving the duration.

**EXERCISE 13.11 •LEVEL:
ADVANCED •MULTIPLE
RESOURCES NO. 2**

- Pick File menu.
- Choose New command.
- Split the screen.
- Create 2w task, "Paint wall."
- Using the task form, assign "Painter." Click OK.
- We need to complete this task sooner so assign a second painter.
- Amend units = "2" (200 percent). Click OK.

What happens to duration and why?

- Create 2w task, "Paint next

wall."

- On bottom screen in the task form, uncheck Effort driven box.
- Assign "Joe the Painter." Click OK.
- Again, we need to complete this sooner.
- Assign "Fred the Painter." Click OK.

The work has doubled rather than the duration halved. We need to halve the work amount on the task.

- Select top view, Pick View menu, pick table, pick work.

- Halve the work on the task.
Click CR.

All is as we would want it; the work has decreased.

MODULE 14: OPTIMIZING THE SCHEDULE

Overview

The optimizing the schedule module introduces how to alter the schedule of a project to make it more efficient, realistic and achievable.

Objectives

The participant will:

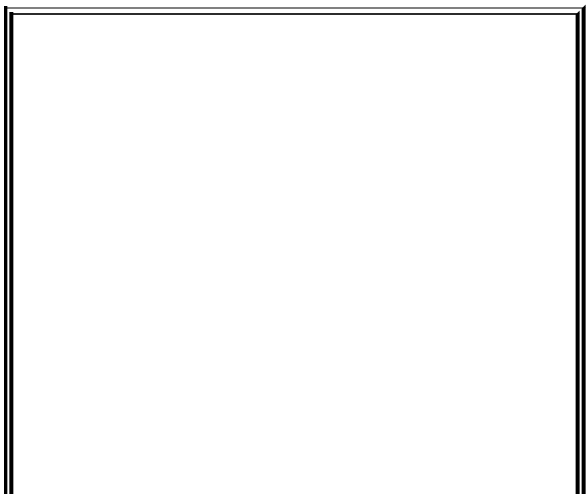
- Be able to maximize a schedule by removing artificial dependencies, by assigning more resources or by adding leads.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise. The level is suitable for basic users.



ACTIVITY

14.1 How to set up tasks to run in parallel.

EXERCISE 14.1 • **LEVEL: BASIC** • **PARALLEL TASKS**

- Open **Golf1F**.
- Remove the dependency between Tasks 6 and 7.
- Remove the dependency between Tasks 7 and 9.
- Set Tasks 6, 7, 9 as successors to Task 5.
- Check the current finish date.
- Remove the dependency

between Tasks 10 and 11.

- Set Task 11 as successor to Task 9.
- Check the current finish date.
- To see this more clearly, split window and display Task PERT view on bottom.

Dialog

Explain that some tasks can be done at the same time as others and can start when one driving task has finished. By setting up tasks in parallel, the end date is often brought forward. What do we do when the end dates do not match the requirements? Change predecessor tasks to undertake some tasks in parallel. Frequently we can eliminate

a finish-to-start dependency between tasks and run the tasks in parallel. In our golf project, the three digging tasks are reasonably independent of one another and so we can undertake them in parallel. The two install tasks are similar.

ACTIVITY

14.2 How to add a resource to reduce the time it takes to do a task.

EXERCISE 14.2 • **LEVEL: BASIC** • **SHORTEN TASK DURATION BY ADDING RESOURCE**

- Assign another "Landscaping crew" to the "Final Clear-up" task, 8 units (800 percent) instead of 4 units (400 percent).
- Check the current finish date.

Dialog

Explain that one means available to a project manager is to add resources to keep to a specific delivery date.

Add extra resources

Sometimes it is possible to add extra resources to a task in order to complete it more quickly.

Dialog

Explain that when tasks are linked, a lag of 0 days is put between them meaning that the successor task starts immediately after the predecessor task finishes. This lag time can be set to another figure to delay the start of the successor task or to have the successor task

starting before the predecessor task finishes. A negative value expresses the overlap lag.

Partial dependencies

This is where the relationship has an additional timing factor to it. Taking an FS (finish-to-start) as an example, a lag is where there is a delay between the predecessor finishing and the successor starting.

A lead is where the

successor
starts before
the
predecessor
finishes.

This can
frequently
be used by a
project
manager to
shorten the
duration of a
project. MS
Project 2000
only
provides the
lag

functionality.
This is no
problem
since a
negative
value for lag
is equivalent
to lead. For
the golf
project, the
seeding
should be
able to start
before the
soil
preparation
is

completed.

ACTIVITY

14.3 How to set an overlapping lag between tasks.

EXERCISE 14.3 • **LEVEL: BASIC** • **OVERLAPPING TASKS**

- Pick View menu.
- Choose PERT Chart.
- Double click on line connecting Tasks 13 and 14.
- Enter lag value of " 3d." Click OK.

(Note the change of start date on Task 14.)

- Pick View menu. Choose Gantt Chart.
- Zoom in to Tasks 13, 14 and note the overlapping of the task bars.

MODULE 15: RESOURCE LEVELING

Overview

The resource leveling module introduces the means to eliminate the overallocation of resources on tasks. This should be done manually by the project manager or MS Project 2000 can do it automatically.

Objectives

The participant will:

- Be able to find resource overallocations.
- Be able to eliminate resource overallocations automatically.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise and whether the level is suitable for basic or advanced users.



ACTIVITY

15.1 How to use the Timescale command.

EXERCISE 15.1 • **LEVEL: BASIC • CHANGE GANTT CHART TO SHOW MONTHS AND WEEKS**

- Open **Golf1G**.
- Pick View menu.
- Choose Gantt Chart.
- Pick Format menu.
- Choose Timescale command.
- Under Major Scale, pick months.

- Under Minor Scale, pick weeks.
- Count = 1.
- Click OK button.

Dialog

Explain that when a project manager is attempting to manually level a project, it is easier to see the overallocations on a more detailed project plan. This is why it is reasonable to change the timescale.

Manual
leveling should
be tried before
automatic

Manual leveling.

leveling Attempt the following to eliminate an overallocated resource.

- Delay start of tasks by entering a delay value
- Increase max. units of resource available
- Assign different resource to task

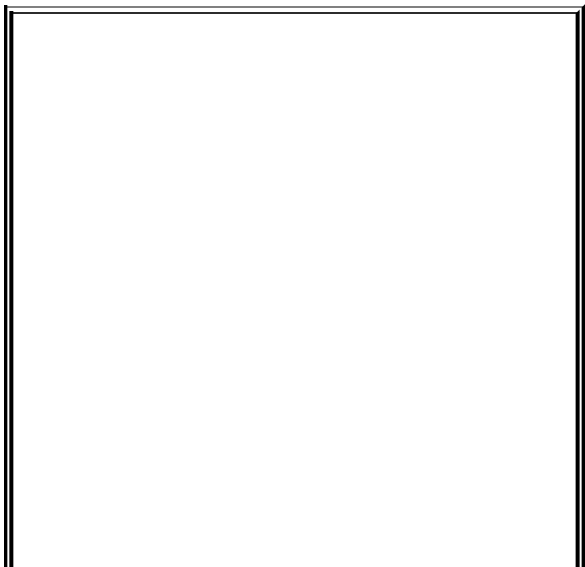
- Change relationships between tasks

- Specify overtime

- Extend working hours

Before we start to examine specific overallocations we need to have a suitable timescale displayed on

the Gantt
chart. We will
display months
and weeks.



ACTIVITY

15.2 How to add the Resource Management toolbar.

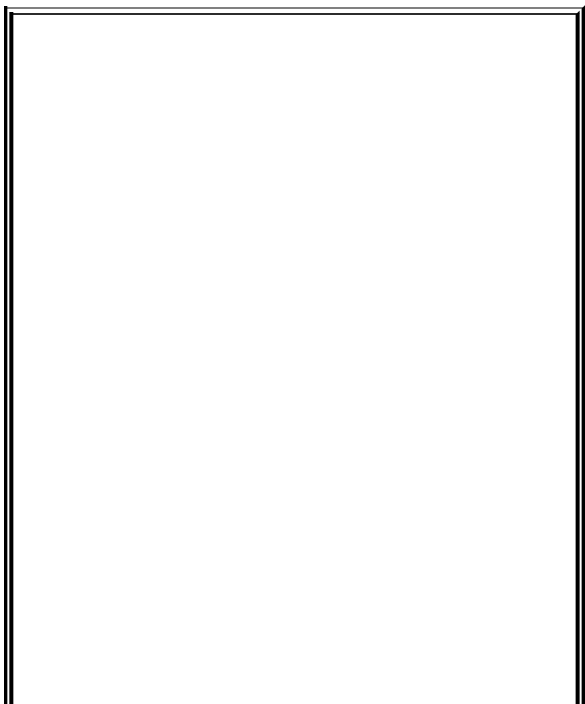
EXERCISE 15.2 • LEVEL: BASIC • ADDING A TOOLBAR

- Pick View menu.
- Choose Toolbars command.
- Select resource management toolbar.
- Look at the new toolbar on the top of your screen.

Dialog

Explain that there is a toolbar specific to managing resources. Now we will

add the resource toolbar to the screen.



ACTIVITY

15.3 How to use the go to overallocation button.

EXERCISE 15.3 • **LEVEL: BASIC** • **FINDING THE OVERALLOCATION**

- Scroll left to display initial task.
- Select first task.
- Pick Window menu. Choose split.
- Select bottom view.
- Pick View menu. Choose Resource Graph.
- Pick Format menu. Choose Details. Select peak units.

- Click on the top view to select the Gantt chart.
- Click on Go to Overallocation button.
- Repeat.

Dialog

Explain that the user can use a combined view to see which tasks are overallocated and who is overallocated. The bottom view will show the resource graph.

ACTIVITY

15.4 How to fix the overallocation.

EXERCISE 15.4 • LEVEL: BASIC • SOLVING THE OVERALLOCATION

- Go to Task 6 (the first overallocation).
- Select bottom view.
- Click on information button. Resource information box appears.
- Set Max. Units = "2" (200 percent) for JCB. Click OK.
- Select "JCB driver" and repeat.
- There is still an overallocation of

1 unit in the first week. How do we fix this?

- Select top view.
- Pick View menu.
- Choose Tables. Select More Tables.
- Pick Delay. Click Apply.
- Apply delay of 7 elapsed days to Task 6.
- Note the overallocation has been eliminated.

Dialog

Explain that how an overallocation is solved is the choice of the project

manager. MS Project cannot make that decision for you. In this exercise, the project manager has chosen to add an additional resource to eliminate the overallocation. The JCB is overallocated because we have two tasks running in parallel. We could move one install task to start after the other but that would increase the timescale by an unacceptable amount. Instead we will allocate more resources to this.

Automatic leveling

MS Project 2000 does this by delaying the start of tasks until the resources are available.

MS Project
2000 looks at
each
overallocated
resource in
turn and
searches for
tasks it can
delay from
those causing
the
overallocation.
MS Project
2000 can't
delay a task if
the task is
constrained

by:

- must start on
- must finish on
- as late as possible
- priority = don't level

A task can't be delayed if an actual start date has been entered, unless task has already been

rescheduled.

For each
potential task
MS Project
2000
considers:

- relationships
- slack time
- start date
- priority
- task

constraints

To decide
which of
several tasks
to delay MS

Project 2000
uses the
following
order of delay
(standard
leveling):

- greatest
slack
- highest
Priority
- highest ID

ACTIVITY

15.5 How to list the tasks associated with the landscaping crew.

15.6 An alternative way to list the tasks associated with the landscaping crew.

EXERCISE 15.5 • **LEVEL: ADVANCED • HOW TO FIND THE TASKS OF THE LANDSCAPING CREW**

- Select top view.
- Pick View menu. Choose Resource Sheet.
- Select bottom view.
- Pick View menu. Choose More

views. Select Resource Form.
Apply.

- Select "Landscape crew" on top view.

**EXERCISE 15.6 • LEVEL:
ADVANCED • HOW TO FIND THE
TASKS OF THE LANDSCAPING
CREW (AGAIN)**

- Select top view.
- Pick View menu. Choose Resource Usage.
- Select bottom view.
- Pick View menu. Choose Gantt Chart.
- Select "Landscape crew" on top

view.

- The tasks of the crew are displayed on the bottom view.
- Scroll through the timescale to display the tasks.

Dialog

Explain that there are various ways to show information so that the project manager can make decisions on how to solve overallocations. For these exercises, we wish to see the tasks that the landscaping crew have been assigned to. We also want to check on the usage of the landscape crew and the tasks they are on.

ACTIVITY

15.7 How to level overallocated resources automatically.

15.8 How to level automatically using priority settings.

EXERCISE 15.7 • **LEVEL: BASIC** • **RESOURCE LEVELING, ALL TASKS SIMILAR**

- Create new project.
- Create 4 tasks, A, B, C and D, 1 day of effort each.
- Select all 4 tasks.
- Pick resource assignment icon and assign "Joe."

- Split Window. Select bottom view.
- Pick View menu. Choose Resource Graph.
- Pick Tools menu. Choose Resource Leveling.
- Pick Level now (entire pool).

**EXERCISE 15.8 • LEVEL: BASIC •
RESOURCE LEVELING,
DIFFERENT PRIORITIES**

- Pick Tools menu. Choose Resource Leveling.
- Select Clear Leveling.
- Pick Tools menu. Choose

Resource Leveling.

- Order = Priority, Standard. Click OK.
- Select Task D. Click on information icon.
- Select General tab, Priority = 550. Click OK.
- Pick Tools menu. Choose Resource Leveling.
- Pick Level now.

Dialog

Explain that automatic leveling is done by MS Project based on when resources are available. If you have constraints in a project, this will

affect how MS Project levels overallocations. If you have priorities set for tasks this will also affect how MS Project levels overallocations.

MODULE 16: USING THE BASELINE

Overview

The using the baseline module introduces the concept of saving a copy of the project plan at the end of the planning period. When the planning is complete, MS Project can take a copy of the start and end dates of each task. This is called the baseline. The baseline is used to check the

progress of the project against the original plan.

Objectives

The participant will:

- Know the difference between baseline, current and actuals.
- Know how to save a baseline.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise and whether the level is suitable for the basic or advanced users.



ACTIVITY

16.1 How to set the baseline for an MS Project file.

EXERCISE 16.1 • **LEVEL: BASIC** • **SETTING THE BASELINE**

- Open **Golf1H**.
- Pick Tools menu.
- Choose Tracking.
- Select Save Baseline.
- Choose for Entire project.
- Click OK button.
- Pick Project menu.

- Choose Project Information command.
- Click statistics button.

Dialog

Explain that a baseline is set only when the user is satisfied that the plan has been developed to a point where it can now be tracked and changed as necessary. There are a few rules to follow regarding baselines and when they should be reset. Note also that MS Project provides a record of baseline information, planned (current) information and in progress (actual) information.

A record of
the task

dates,

Baseline duration, work and costs, as originally planned.

The baseline

can be set more than

once if the schedule is

deviating too greatly from

the plan or if a number of

additional tasks have

been

**Re-setting
baseline**

identified.

There are three sets of dates identified by MS Project 2000 when using the baseline:

Current

The planned scheduled start and end dates of each task. These may change with each amendment you make to the plan.

The original start and end

Baseline dates of each task as saved in the baseline.

The start date of tasks in progress or the start and end dates of tasks that have been completed.

Actual

MODULE 17: UPDATING TO REFLECT ACTUAL PROGRESS

Overview

This module introduces how track the project using the status of each task.

Objectives

The participant will:

- Be able to track the progress of the plan.

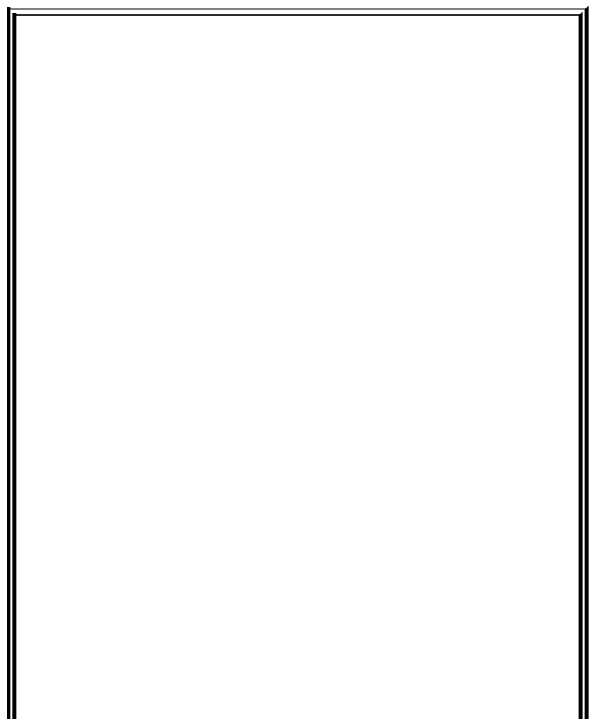
Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activities in this module cite the aims of each exercise.

The level is suitable for basic users.



ACTIVITY

17.1 How to add the tracking toolbar and set the current date.

EXERCISE 17.1 • **LEVEL: BASIC** • **VIEW COMBINATION FOR UPDATING AND SETTING DATE**

- Open **Golf2A**.
- Pick View menu.
- Select Gantt Chart.
- Point cursor at a toolbar, click the *right-hand* mouse button, pick Tracking.
- Pick Project menu.
- Choose Project Information

command.

- Set current date = "June 21, 2000."
- Click OK button.

Dialog

Explain that the tracking toolbar is a handy tool to use when tracking the progress of a plan. To demonstrate the various ways to track a project, we will update the status of this project as of June 21, 2000. There are two approaches to monitoring project status as work progresses:

- update on a task-by-task basis using task status
- update on a resource-by-

resource basis using resource work

In MS Project 2000 either method may be used, although task status is much easier since MS Project 2000 provides a number of tools to assist. Regardless of which method is used a basic project administration system is required in order to gather the data needed.

The status information you need at each update time includes:

- the dates each task actually started or finished
- how long the task took
- how much work remains to be completed at the time of your

status check

You can update all dates manually or let MS Project 2000 automatically update for you. To update manually, an appropriate view to use is the Gantt chart with the tracking table.

ACTIVITY

17.2 How to use the mouse and cursor to update the status of the project.

EXERCISE 17.2 • **LEVEL: BASIC** • **UPDATE PROJECT STATUS USING THE MOUSE**

*Status information for updating on
June 21*

Task 2 is complete

Task 4 is complete and finished
2 days early

Task 5 is half done

- Point to LHS of bar for Task 2 (cursor changes to %) and drag to right until complete.
- Point to RHS of bar for Task 4 (cursor changes to vertical bar) and drag to left until approx 3.5w (zoom-in if necessary).
- Point to LHS and drag to right until complete.
- Point to LHS of bar for Task 5 and drag to right until half complete.

- Pick View menu, Pick more views, pick Tracking Gantt. Click Apply.

Dialog

Explain that the user can update the project plan directly onto the Gantt chart by using the cursor. The cursor changes shape depending on how it is placed over the bars showing on the calendar section of the Gantt chart.



ACTIVITY

17.3 How to use the tracking table to update the status of the project.

EXERCISE 17.3 • **LEVEL: BASIC** • **UPDATE PROJECT STATUS USING TABLES**

Status information for updating on August 2

Task 5 took 6 weeks instead of 5 to complete

Tasks 6, 7, 9, 10 are all progressing as planned.

Task 11 only started on July 15 due to bad weather

- Set current date = Aug 2, 2000.
- Pick View menu.
- Choose More views.
- Select Task sheet.
- Pick View menu.
- Choose Table.
- Select Tracking.
- Enter actual durations, 5 =

30d, 6 = 14d, 7 = 9d, 9 = 5d, 10 = 9d. Note the other column values are updated.

- Task 11, actual start = July 15.
- Pick View menu. Choose Gantt chart. Click OK button.

Dialog

Explain that there are difficulties in attempting to update the project by only using the cursor. By using the

tracking table to update the project, the user can alter task durations and start dates by inserting specific data into the table.



ACTIVITY

17.4 How to automatically update the status of the project.

EXERCISE 17.4 • **LEVEL: BASIC** • **UPDATE PROJECT STATUS AUTOMATICALLY**

*Status information for updating on
August 30*

Tasks 6, 7, 10 completed on
schedule.

Task 11 still having problems.

The project is now getting
seriously behind schedule and
we need to take some remedial

action.

- Set current date = Aug 30, 2000.
- Select 6, 7, 10.
- Pick Tools menu.
- Choose Tracking.
- Select Update Project.
- Update for selected tasks.
- Click OK button.

- Select Task 11.
- Pick Tools menu.
- Choose Tracking.
- Select Update Tasks.
- Update Actual dur = 30d,
Rem dur = 5d.

Dialog

Explain that the user can update the project status automatically in MS Project. However, the user has a choice:

to update the entire project or
to update selected tasks only.
This exercise shows how to
update selected tasks only.

MODULE 18: MULTIPLE PROJECTS

Overview

The multiple projects module introduces how to manage multiple projects with MS Project 2000.

Objectives

The participant will:

- Know how to use a resource pool to manage multiple projects.
- Know how to merge a number of projects into one MS Project file.

Preparation

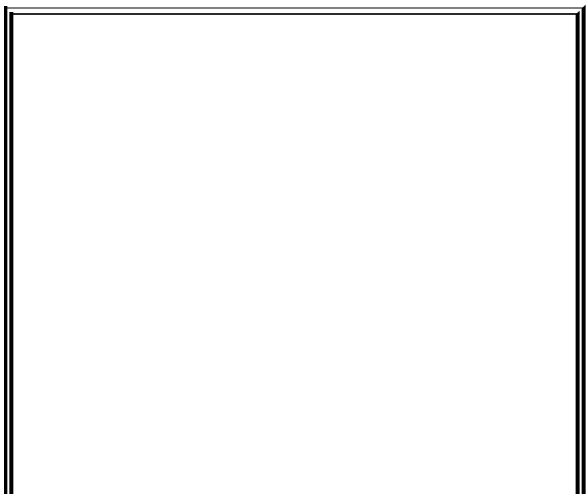
Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Resource pool

This is the total of all resources in a project. To use the concept of a resource pool we firstly create a new project and define all resources in this. We normally do not define any tasks in the project. We then create our projects as normal but tell MS Project 2000 to take the resource definitions from the resource project.

Presentation

The activities in this module cite the aims of each exercise. The level is suitable for basic users.



ACTIVITY

18.1 How to set up a resource pool for use with more than one project.

EXERCISE 18.1 • **LEVEL: BASIC** • **USING A RESOURCE POOL**

- Pick File menu. Choose New.
- Pick View menu. Choose Resource Sheet.
- Enter Joe, Fred, Mary, Anne.
- Pick File menu. Choose Save As, Filename = Resource.
- Pick File menu. Choose New.
- Create task1, task2, task3.

- Pick Tools menu. Choose Resources. Select Share Resources.
- Use Resources from Resources project. Click OK button.
- Assign Joe to task1. Assign Mary to task2. Assign Fred to task3.
- Pick File menu. Choose New.
- Create taskA, taskB, taskC.
- Pick Tools menu. Choose Resources. Select Share Resources.
- Use Resources from Resources project. Click OK button.

- Assign Fred to taskA. Assign Mary to taskB. Assign Anne to taskC.
- Pick Window menu. Choose Resources project.
- Pick View menu. Choose Resource Sheet.
- Note the overallocated resources in red.
- Pick Window menu. Choose Split.
- Choose overallocated resource.
- Resource form is displayed on bottom.

Dialog

Explain that if you have a limited number of resources and a number of projects that they are working on, you can set up a central pool from which to draw those resources. The command, Share Resources, allows you to pull resources from another MS Project file. This is one method to track how your resources are overallocated due to them working on a number of projects at the same time. This method also allows you to resolve those overallocations.

ACTIVITY

18.2 How to place more than one project on a new MS Project file.

EXERCISE 18.2 • **LEVEL: BASIC** • **REPORTING SEPARATE PROJECTS TOGETHER**

- Open all projects to be reported on.
- Pick Window menu.
- Choose New window.
- Select required projects.
- Click OK button.

Dialog

Explain that there times when you may want to consolidate a number of projects in one MS Project file. This exercise explains how this is done. Note that any changes made to this file will be reflected in the original file.

We can view together any projects we wish without having to create a link between them. Open each project separately and use the

**View
separate**

projects New Window command to merge the views. We can use this to print reports from more than one project at a time.

ACTIVITY

18.3 How to add sub-projects to an MS Project file.

EXERCISE 18.3 • **LEVEL: BASIC** • **REMOVE PHASE I INTO A SUB-PROJECT**

- Open **Golf2A**.
- Select the detailed tasks in Phase 1 (4 - 7).
- Click on cut icon to remove complete tasks.
- Click on New file icon.
- Click on Paste icon.
- Pick File menu. Choose Save,

Filename = Phase 1.

- Pick Window menu. Choose **Golf2A**.
- Delete row three and choose the insert project command.
- Select Phase 1 and insert.
- Change duration of task in project Phase 1 and see task Phase 1 duration change.

Dialog

Explain that there may be some instances when a user wishes to insert a project or part of a project into an MS Project file. This exercise explains how this is done. Note, however, that the cut and paste

functions did not retain the start dates originally established in the Golf2A file. This method can be used when setting up project templates, for example, in the localization industry.

The concept behind this is almost identical to that of outlining. Instead of a task in a project being a summary task, as in outlining, a

Link sub- projects

task is an external separate project. The start and end dates of this sub-project task are set by the external project. This gives you the ability to have many sub-projects and to summarize them in one or more parent

projects.

You can design your projects in this format from the beginning or, more commonly, you may find that your project is getting too big to manage and you need to divide it during execution.

Use this
method to set
up project
templates.

MODULE 19: USING REPORTS

Overview

The Using Reports Module introduces how to generate reports using the Reports function within MS Project 2000.

Objectives

The participant will:

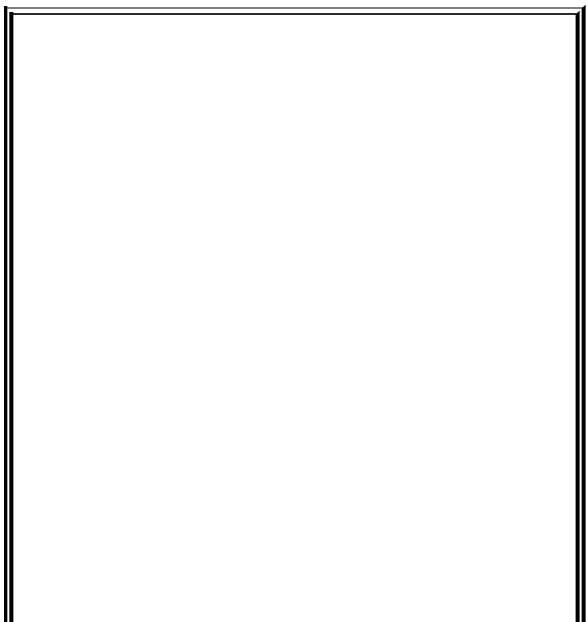
- Be able to create and print a variety of reports using the Reports function.

Preparation

Review the contents of this module and do each of the exercises at least once to familiarize yourself with the logic and outcome.

Presentation

The activity in this module cites the aim of the exercise and that the level is suitable for basic users.



ACTIVITY

19.1 How to open and view each report within the Reports function.

EXERCISE 19.1 • **LEVEL: BASIC** • **EXAMINING REPORT TYPES**

- Pick View menu.
- Choose Reports.
- Five (5) groups of predefined reports are available.
- Review each report.

Dialog

Explain that MS Project 2000 has a series of pre-formatted reports in it, that will print specific information in a

table format. Each report only provides certain information that may be used, at certain times in the project, to report on specific data. Note that the you can create your own reports by using the customize reports function or use the filter command to change the information showing on the computer display screen.

REFERENCES AND FURTHER READING

[References](#)

[Further reading](#)

References

Amundsen, Roald (1912). *The South Pole*. John Murray

Boehm, Barry (1981). *Software Engineering Economics*.
Prentice Hall

Brooks, F.P. (1975). *The Mythical Man-Month*, Addison-Wesley

Brooks, F.P. (1987). *No Silver Bullet: Essence and Accidents in Software Engineering*. IEEE

Computer, April

Catton, Bruce (1960). *Pictorial History of the Civil War*.

American Heritage Publishing Co.

Catton, Bruce (1961). *The Coming Fury*. Washington Square Press

Catton, Bruce (1963). *Terrible Swift Sword*. Washington Square Press

Catton, Bruce (1965). *Never Call Retreat*. Washington Square Press

Clausewitz (1984). *On War*.
Penguin Classics

DeMarco, Tom (1978).
*Structured Analysis and
Systems Specification*. Prentice
Hall

DeMarco, Tom and Lister,
Timothy (1987). *PeopleWare*.
Dorset House Publishing

Eberts, Jake and Ilott, Terry
(1990). *My Indecision is Final*.
Faber and Faber

Fisher, Roger and Ury, William
(1981). *Getting to Yes*.

Hutchinson Business

Griffiths, Trevor (1986).
Judgement Over the Dead.
Verso

Hampton, Henry and Frayer,
Steve (1990). *Voices of
Freedom.* Bantam

Huntford, Roland (1985).
Shackleton. Hodder and
Stoughton

Huntford, Roland (1985). *The
Last Place on Earth.* Pan Books

Huntford, Roland (1993). *Scott*

and Amundsen. Wiedenfeld and
Nicholson

Macdonald, John (1984). *Great
Battlefields of the World.*
Michael Joseph

Macdonald, Lyn (1978). *They
Called It Passchendaele.*
Macmillan

Macdonald, Lyn (1983).
Somme. Michael Joseph

McMurtry, Larry (1990).
Lonesome Dove. Pan Books

Milne, A.A. (1923). *The House*

at Pooh Corner. Methuen

O'Connell, Fergus (1999). *How to Run Successful High Tech Project-Based Organizations.* Artech House

Puttnam, Laurence H. and Myers, Ware (1992). *Measures for Excellence.* Yourdon Press
Rothery, Brian (1991). *ISO 9000.* Gower

Terraine, John (1977). *The Road To Passchendaele.* Leo Cooper

Further reading

Ballard, Robert (1987). *The Discovery of the Titanic*. Hodder and Stoughton

Boorman, John (1985). *Money Into Light*, Faber and Faber

Bradley, Ken (1993). *PRINCE: A Practical Handbook*. Butterworth-Heinemann

Bruce, Phillip and Peterson, Sam M. (1982). *The Software Development Project*. Wiley-

Interscience

Cherry-Garrard, Apsley (1983). *The Worst Journey in the World*. Penguin Books

Conrad, Joseph (1912).
Recollections on the loss of the
Titanic. *New English Review*,
May

Crosby, Phillip (1979). *Quality is Free*. McGraw-Hill

Davie, Michael (1986). *The Titanic*. The Bodley Head

DeMarco, Tom (1982).

Controlling Software Projects.
Yourdon Press

Frey, James N. (1987). *How to Write a Damn Good Novel.* St. Martin's Press

Fussell, Paul (1975). *The Great War and Modern Memory.*
Oxford University Press

Giles, John (1977). *The Somme Then And Now.* Bailey Brothers and Swinfen

Gliddon, Gerald (1987). *When the Barrage Lifts.* Gliddon Books

Gracie, Colonel Archibald
(1991). *Titanic*. The Blackstaff
Press

Huntford, Roland (1987). *The
Amundsen Photographs*.
Hodder and Stoughton

International Standards
Organization (1987). *ISO 9001
Quality Systems Model for
Quality Assurance in Design,
Development, Production,
Installation and Servicing*. First
Edition

Limb, Sue and Cordingley,
Patrick (1982). *Captain Oates*

Soldier and Explorer.

Batsford

Lord, Walter (1981). *A Night To Remember*. Penguin Books

Marcus, Geoffrey (1969). *The Maiden Voyage*. Allen & Unwin

Mear, Roger and Swan, Robert (1987). *In The Footsteps of Scott*. Jonathan Cape

Middlebrook, Martin (1971). *The First Day on the Somme*. Allen Lane

Orr, Philip (1987). *The Road to*

the Somme. The Blackstaff Press

Ponting, Herbert (1921). *The Great White South.* Duckworth

Ralling, Christopher (1983). *Shackleton.* BBC

Savours, Ann (1974).
Scott's Last Voyage Through
the Antarctic Camera of
Herbert Ponting. Sidgwick and
Jackson

Scott, Robert Falcon (1914).
Scott's Last Expedition (2 vols).
Smith, Elder & Co.

Scott, Robert Falcon (1983). *Scott's Last Expedition*.
Methuen

Seymour, William (1982). *Yours to Reason Why* Decision in
Battle. Sidgwick and Jackson

The Times History of the War

Tolkien, J.R.R. (1968). *The Lord of the Rings*. Allen & Unwin

Wade, Wyn Craig (1980). *The Titanic End Of A Dream*,
Penguin Books

Ward, Geoffrey C., Burns, Ric

and Burns, Ken (1990). *The Civil War*. American Documentaries

Watson, Arnold and Watson, Betty (1984). *Roster of Valor The Titanic Halifax Legacy*. 7 C'S Press

Weinberg, Gerald M. (1975). *An Introduction to General Systems Thinking*. John Wiley & Sons

Wiener, Lauren Ruth (1993). *Digital Woes Why We Should Not Depend on Software*. Addison-Wesley

Winkler, John (1989). *Winning Sales and Marketing Tactics*.
Butterworth Heinemann

Yule, Andrew (1988). *David Puttnam: The Story So Far*.
Mainstream Publishing Co.